2014

Service composition based on SIP peer-to-peer networks

Lehmann, Armin

Service composition based on SIP peer-to-peer networks

by

Armin Lehmann

A thesis submitted to Plymouth University

in partial fulfilment for the degree of

Doctor of Philosophy

School of Computing and Mathematics

In collaboration with

Darmstadt Node of the CSCAN Network

March 2014

Service composition based on SIP peer-to-peer networks

Armin Lehmann

Abstract

Today the telecommunication market is faced with the situation that customers are requesting for new telecommunication services, especially value added services. The concept of Next Generation Networks (NGN) seems to be a solution for this, so this concept finds its way into the telecommunication area. These customer expectations have emerged in the context of NGN and the associated migration of the telecommunication networks from traditional circuit-switched towards packet-switched networks.

One fundamental aspect of the NGN concept is to outsource the intelligence of services from the switching plane onto separated Service Delivery Platforms using SIP (Session Initiation Protocol) to provide the required signalling functionality. Caused by this migration process towards NGN SIP has appeared as the major signalling protocol for IP (Internet Protocol) based NGN. This will lead in contrast to ISDN (Integrated Services Digital Network) and IN (Intelligent Network) to significantly lower dependences among the network and services and enables to implement new services much easier and faster. In addition, further concepts from the IT (Information Technology) namely SOA (Service-Oriented Architecture) have largely influenced the telecommunication sector forced by amalgamation of IT and telecommunications. The benefit of applying SOA in telecommunication services is the acceleration of service creation and delivery. Main features of the SOA are that services are reusable, discoverable combinable and independently accessible from any location. Integration of those features offers a broader flexibility and efficiency for varying demands on services.

This thesis proposes a novel framework for service provisioning and composition in SIP-based peer-to-peer networks applying the principles of SOA. One key contribution of the framework is the approach to enable the provisioning and composition of services which is performed by applying SIP. Based on this, the framework provides a flexible and fast way to request the creation for composite services. Furthermore the framework enables to request and combine multimodal value-added services, which means that they are no longer limited regarding media types such as audio, video and text. The proposed framework has been validated by a prototype implementation.

Contents

List of Figures

List of Tables

Acknowledgements

In the first place I wish to express my sincere thanks to my supervisors Prof. Woldemar Fuhrmann and Dr. Bogdan Ghita for their kind and helpful support and guidance throughout this research.

I would like to express the deepest appreciation to my supervisor Prof. Ulrich Trick for the continuous support throughout my time as a member of the Research Group for Telecommunication Networks at University of Applied Sciences Frankfurt. His guidance helped me in all the time of research and writing of this thesis. This research would probably never have been performed without his encouragement.

Many valuable ideas and suggestions regarding this research arose from innumerable discussions with my colleagues and friends at the Research Group for Telecommunication Networks at University of Applied Sciences Frankfurt. I wish to thank all current and former members of this group for the great inspiration that I have experienced.

Warm thanks go to the members of both the graduate school and the CSCAN Network at Plymouth University, and special thanks go to the members of the CSCAN Darmstadt node for their experienced support in academic aspects.

I wish to thank my family and friends for their non-technical support throughout the whole time of the research.

Finally, my biggest thanks belong to my daughter Lara, for her great understanding and patience throughout the entire time of this research.

Author's Declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Graduate Committee.

Relevant scientific seminars and conferences were regularly attended at which work was often presented, and several papers were prepared for publication.

Word count of main body of thesis: 34.545

Signed _____

Date _____

1 Introduction

“Business, when boiled down to the essentials, is all about providing a customer with a product or service at a price that the customer is prepared to pay and at a price that covers the provider’s costs.” (Salisbury, 2007)

Over the last few years, the telecommunication market is faced with the situation, that the customers are not willing to pay the old prices any more for basic telecommunication services such as telephony or SMS (Short Message Service). This leads to new challenges in the telecommunication market, which results in the need of a more cost efficient service creation and delivery. The main factors for this unwillingness to pay the old prices are that new service providers entered the telecommunication market, because of the deregulation. Also alternative technologies, principally based on or around IP (Internet Protocol), are supporting legacy services at lower costs than based on traditional technologies and provide possibilities for new services that cannot be delivered by traditional technologies (Salisbury, 2007).

The heterogeneity of the old telecommunication infrastructure, the growing competition and the drop in revenue can be regarded as the primary threats to the telecommunications industry. Telecommunication operators were forced to rethink their business models and to convert their infrastructure to a fully IP-based platform – the Next Generation Network, because of the aim to reduce costs and to create new sources of income (T-Systems, 2007).

One crucial reason for deployment of NGN (Next Generation Networks) is the separation of transport and service functionalities, which is one of its key features

according to (ITU-T Y.2001, 2004) and (ETSI TR 180 000, 2006). From a service provider point of view this is a great advantage in contrast to traditional telecommunication networks where the networks were more or less implemented specifically for telephony services. NGN provides one packet-based transport network for a broad range of services with different demands for bandwidth which depend for example on the utilised media type (such as video, voice or text). Concerning the migration towards NGN, SIP (Session Initiation Protocol) has appeared as the major signalling protocol to facilitate its integration and transition from traditional telecommunication networks. As the name implies, SIP supports the establishment, change, and termination of multimedia sessions, such as audio and video telephone calls. Once a session has been initiated, the media data are exchanged directly via the packet-switched transport network. The NGN service infrastructure typically based on the Session Initiation Protocol is not required to be involved in the media data transfer and, hence, its dimensioning and layout is completely media-independent, which is a strong benefit compared to PSTN (Weber, 2012). Furthermore, the value-added services for which the customers are willing to pay are deployed on independent servers – the application servers (AS). This offers the possibility to develop and provide new services, especially value-added services (VAS), independent of the underlying networks.

Further benefits can be achieved by following recommendations given by concepts from the IT (Information Technology) sector, where the SOA (Service-Oriented Architecture) has largely influenced telecommunication services area. The main features of SOA are that services should be reusable for creation of new services by combining them; services also should be independently accessible from any location

which implies that they have to be discoverable. This results in higher efficiency and flexibility. Beyond that, telecommunication services are focused on human senses (visual, aural, olfactory, taste, and tactile), nowadays only on those senses which lend themselves to “long-distance” communication e.g. sight and sound (Salisbury, 2006). In future, the support of all human senses by service providers will offer a competitive advantage when entering into the market and introduce new value-added services which combine multiple human senses (Woyczechowski, 2008).

Although in NGN most services are provided by application servers, there is no uniform standard, as provided services are combined to new services. These NGN-based solutions implemented as IMS (IP Multimedia Subsystem) are limited on studies or non-standardised approaches, such as SCIM (Service Capability Interaction Manager) or Service Brokers. Service Delivery Platforms offer new opportunity in this regard, because they are decoupled from the underlying network. Mostly the service delivery platform solutions are based on specific implementations with centralised control, which is single point of failure. A peer-to-peer-based approach could be another solution, such as eliminating single point of failure, better use of (distributed) resources, and a better scalability, in contrast to a centralised control.

This research work has been dedicated to find and describe a novel approach for provisioning and combining services in NGN following the concepts given by SOA. The research work should enable any user to provide services on their own to any other user. The aims and objectives of this research are presented in section 1.1, followed by an outline on the thesis structure in section 1.2.

1.1 Aims and objectives

The aim of this research is to propose a framework that allows combining, discovering and providing value-added services in a SIP-based peer-to-peer network involving the concepts of NGN and SOA. It should be possible to provide value-added services for anyone. This means, that not only telecom operators are enabled to do this. Also ordinary users are facilitated to provide their own services. So anybody can act as a service provider.

The main objectives of this research can be outlined as follows.

1. To analyse the existing approaches of SIP-based peer-to-peer networks regarding their advantages and disadvantages. Subsequently, to specify the basic architecture of the framework resting upon a SIP peer-to-peer infrastructure.
2. To work out concepts and principles regarding NGN and SOA the framework has to follow. Furthermore to analyse and evaluate different service composition concepts resulting in requirements for optimised concept.
3. To define the architecture and associated functionality of the proposed framework, this is for optimised service composition in SIP-based peer-to-peer networks.
4. To examine different service interface descriptions, resulting in a proposed service description language.
5. To specify an adequate algorithm to enable the creation of service compositions based on the proposed service description language.

6. To suggest recommendations and restrictions for the framework, so that services will not interact in unwanted ways.
7. To implement and evaluate the proposed architecture for optimised service composition using a prototype implementation.

The order of objectives declared above corresponds to the general structure of this thesis as presented within the following section.

1.2 Thesis structure

Chapter 2 outlines the concept of NGN and gives an overview of the SIP functionality principles and its widespread utilisation in NGN-based architecture. Beside these principles of peer-to-peer networks are illustrated and furthermore different ways to implement SIP-based peer-to-peer networks are discussed.

Chapter 3 addresses the notion of services within the telecommunication area. It starts by illustrating the concepts of Service-Oriented Architecture and Service Delivery Platforms. Furthermore, the service provisioning in IMS and different service composition concepts such as in NGSON (Next Generation Service Overlay Networks) are presented. The main outcome of this chapter is a set of well-founded requirements for the proposed approach.

Chapter 4 introduces the proposed framework for service composition. The resulting framework as a main outcome of the research work follows the requirements defined in chapter 3 describing the overall framework, its components and their functionality.

Chapter 5 describes the concept of service description language, given its major role in service composition. The chapter critically analyses a number of service description

languages, which leads to a novel SDL (Service Description Language), which is identified to fulfil all emerged characteristics to fully describe the service functionality and facilitates it for service composition. Furthermore an algorithm for service composition utilising the service description language is presented.

Chapter 6 discusses further recommendations and restrictions for service creation and interaction. Service design recommendations are illustrated which will help to reduce unwanted service interactions. Also general design recommendations for services are depicted.

Chapter 7 introduces a proof of concept implementation and focuses on the evaluation of the framework based on a set of identified criteria. The research prototype is subsequently utilised to evaluate most parts of the framework through a number of scenarios. Furthermore the algorithm which performs the service composition is tested and evaluated separately. Finally, the whole framework is evaluated regarding the identified criteria.

Chapter 8 presents the main conclusions of this research work listing its achievements and its limitations. The chapter also includes a section with a number of possible avenues to explore for future work.

The thesis is provided with a number of appendices in support of the presented research.

2 SIP (Session Initiation Protocol) - based telecommunication infrastructures

This chapter introduces the fundamental telecommunication architecture that this research is based on. The discussion starts with an overview of the NGN concept and its general architecture (section 2.1), then it focuses on SIP and its functionality (section 2.2); finally, section 2.3 provides an overview of different existing peer-to-peer models especially those combined with SIP or using SIP.

2.1 The concept of Next Generation Networks (NGN)

The concept of NGN appeared in the mid-1990s has recently become widely accepted within the field of both, fixed and mobile telecommunication networks. The conversion of telecommunication networks from traditional circuit-switched technologies (such as ISDN (Integrated Services Digital Network)) towards packet-switched NGN currently takes place. The concept of NGN to face the emerging situations in telecommunications characterised by a lot of different factors (Cochennec, 2002):

- strong open competition between operators,
- explosion of data traffic due to the use of the Internet,
- strong demand from users for new multimedia services, and
- increasing demand from users for general mobility.

The ITU-T (International Telecommunication Union – Telecommunication Standardization Sector) started its research work concerning NGN in the year 2000. In (ITU-T Y.2001, 2004) a NGN has been defined as:

A packet-based network able to provide telecommunication services and able to make use of multiple broadband, QoS-enabled (Quality of Service) transport technologies and in which service-related functions are independent from underlying transport-related technologies. It enables unfettered access for users to networks and to competing service providers and/or services of their choice. It supports generalized mobility which will allow consistent and ubiquitous provision of services to users.

NGN research and standardisation work performed by both ETSI (European Telecommunication Standards Institute) and ITU-T has been synchronised regarding all relevant aspects. Hence, according to (ITU-T Y.2001, 2004), (ETSI TR 180 000, 2006) and (Trick and Weber, 2004) the term NGN stands for a telecommunication network concept that can be characterised by the following key features.

1. Packet-based data transport
2. Quality of Service support
3. Applicability for arbitrary services
4. Separation of call/service control and media data transport
5. Capable for the integration of any existing, important telecommunication network, especially access networks
6. Application Server support
7. Support for multimedia services
8. High bit rates
9. Overall unified network management
10. Mobility support
11. Integrated security functions
12. Service-appropriate charging

13. Scalability
14. Unrestricted access for users to different networks and service providers
15. Consideration of obligatory legal requirements (such as lawful interception and emergency calling features)

The PhD research is tied to the NGN concept, especially the points 1, 3, 4, 6, 7, 13 and 14, because they are related to services. Services and service-oriented architectures are in the main focus of this work.

According to (ITU-T Y.2012, 2010), a NGN can generally be divided into two strata; the service stratum and the transport stratum. The transport stratum is composed of transport functions, which provide connectivity for all components and physically separated functions within NGN, and further transport control functions, such as admission control and mobility management functions. The service stratum that is located above the transport stratum consists of two functional groupings, the service control and content delivery functions and the application support functions.

The former functions include features such as registration, and authentication, and authorisation as well as features to deliver services to end-users provided by utilising the application support and service control functions. The application and service support functions include features such as gateway, registration, authentication and authorisation at the application level. These two functions work in conjunction with the service control functions to provide end users and applications with the NGN services they request.

According to (ITU-T Y.2012, 2010) applications are outside of the NGN scope. This might be problematic, because current changes of the telecommunication networks will lead to significant interest in applications. Applications will be one of the main

revenue streams in future telecommunication networks. Other researches on NGN will not leave applications outside the NGN's scope, according to (Trick and Weber, 2009). The end users equipment, e.g. telephones, can be connected directly or via diverse access technologies, therefore media and signalling gateways can be applied. Service requests are typically handled by call servers (CS). Application servers can be involved in order to provide advances services, so-called value-added services. Also the NGN offers access to other networks such as Internet and circuit-switched and/or packet-switched telecommunication networks by gateways.

2.2 SIP architecture and functionality

The Session Initiation Protocol has been defined by IETF (Internet Engineering Task Force) in (IETF RFC 3261, 2002) and numerous extensions to the SIP base protocol have since been defined in several further RFCs (Request For Comments). As an application layer protocol, SIP messages are carried over IP.

SIP intends to initiate and manage communication sessions within the area of VoIP MoIP (Multimedia over IP), typically dealing with VoIP (Voice over IP). SIP implements several mechanisms to provide connection-oriented and reliable service functionality and offers also advanced functions (such as Instant Message Transmission). In UMTS (Universal Mobile Telecommunication System) Release 5 (ETSI Tdoc RP 030375, 2003), SIP was chosen in order to provide multimedia communications. The 3GPP (Third Generation Partnership Project) defined an abstract architecture called IP Multimedia Subsystem (IMS) (3GPP TS 23.228, 2006), which is "a global, access-independent and standard-based IP connectivity and service control architecture that enables various types of multimedia services to end-users

using common Internet-based protocols" (Poikselkä and Mayer, 2009). Originally the IMS was an approach to deliver IP-based services over GPRS (General Packet Radio Service).

The IMS represents an extended SIP service infrastructure with well-defined interfaces providing interconnection capabilities (e.g. for the integration of service delivery platforms). The IMS is based on SIP as standardised by IETF. SIP was originally defined for Internet use. The protocol and its related service architecture have been designed for the use in a public IP environment, consisting of a multiplicity of interconnected IP transport networks operated by mutually independent providers (IETF RFC 3261, 2002).

2.2.1 Basic SIP functionality

The basic standard for SIP is RFC 3261 (IETF RFC 3261, 2002), where the main SIP functionality is defined. The SIP architecture includes the following entities as given by RFC 3261: SIP User Agent, SIP Proxy and SIP Registrar server and Location server. In a SIP-based telecommunication infrastructure user end systems are called SIP User Agents. They provide to establish and manage SIP sessions for instance in VoIP. Any SIP entity has to be bound to an IP address, but due to the volatility and lack of user-friendliness of IP addressing, every SIP subscriber is provided with a permanent SIP URI (Uniform Resource Identifier) (e.g. sip:username @provider.co.uk) by its service provider. The service provider runs a SIP service infrastructure, which consists of different logical server entities (such as SIP Proxy servers, SIP Registrar servers, and Location servers). Multiple SIP servers can be integrated in one IP host or they can be realised as distributed servers. As defined in

(IETF RFC 3261, 2002) User Agents are enabled to communicate in peer-to-peer fashion, as long as they have knowledge regarding the temporary SIP URI of the corresponding communication partner they can establish a session with SIP. After the session initiation is completed, the involved User Agents establish a logical connection-oriented communication state (referred to as a SIP dialog) between them, as the end systems are ready to exchange media data of arbitrary nature (such as VoIP and/or video data flows) by making use of any appropriate transport protocol, which is typically RTP (Real-time Transport Protocol) (IETF RFC 3550, 2003) for voice and video flows. The User Agents negotiate the media to be exchanged and the associated codec's during the session establishment phase by exchanging SDP (Session Description Protocol) media descriptions (IETF RFC 4566, 2006) carried within the SIP messages. The User Agents exchange the media data packets in a peer-to-peer manner over the IP network.

2.2.2 Common utilisation of SIP in telecommunication

SIP has become the de-facto standard in IP-based telecommunication networks that follow the NGN concept. 3GPP has defined IMS as core for multimedia communication based on IP for mobile communication in UMTS Release 5. IMS is using SIP as its general signalling protocol besides the Diameter protocol (IETF RFC 6733, 2012), which is used to provide its underlying Authentication, Authorisation, and Accounting (AAA) framework. ETSI, as one of six Organisational Partners of the 3GPP, has established the ETSI TISPAN (Telecommunications & Internet converged Services and Protocols for Advanced Networking) standardisation body in 2003. The

focus of TISPAN is on defining NGN in Europe. TISPAN has chosen IMS as one subsystem within the Service Stratum in NGN (ETSI TR 180 000, 2006).

A SIP-based is typically based on an IP transport infrastructure, with SIP playing the role of the signalling protocol within the service stratum.

The generic structure of a SIP-based NGN is shown in Figure 2.1. A SIP service provider operates the core infrastructure. This infrastructure contains a centralised call server (CS), whose functionality is provided by SIP Proxy and SIP Registrar servers. These servers rely on Location servers to find the matching temporary SIP URIs for given permanent SIP URIs. Application servers (AS) and media servers (MS) provide value-added services to the end users. A special implementation of such AS and MS is a Conference server. Access to/from circuit-switched networks will be enabled by Media and Signalling Gateways. The SIP User Agents consist of the end user equipment. A special implementation of such AS and MS is a Conference server. Access to/from circuit-switched networks will be enabled by Media and Signalling Gateways. The SIP User Agents consist of the end user equipment.

If required, SIP signalling and media streams can be routed in parallel via trusted intermediate service layer network elements, such as by the SIP service provider. This is potentially useful for the consideration of certain legal requirements such as lawful interception, for the interconnection with other NGN providers, and for NAPT (Network Address and Port Translation) traversal. SIP Back-to-Back User Agents (B2BUA) are used for this purpose, implemented in network elements coming in different flavours such as Session Border Controllers (SBC) or Application Layer Gateways (ALG).

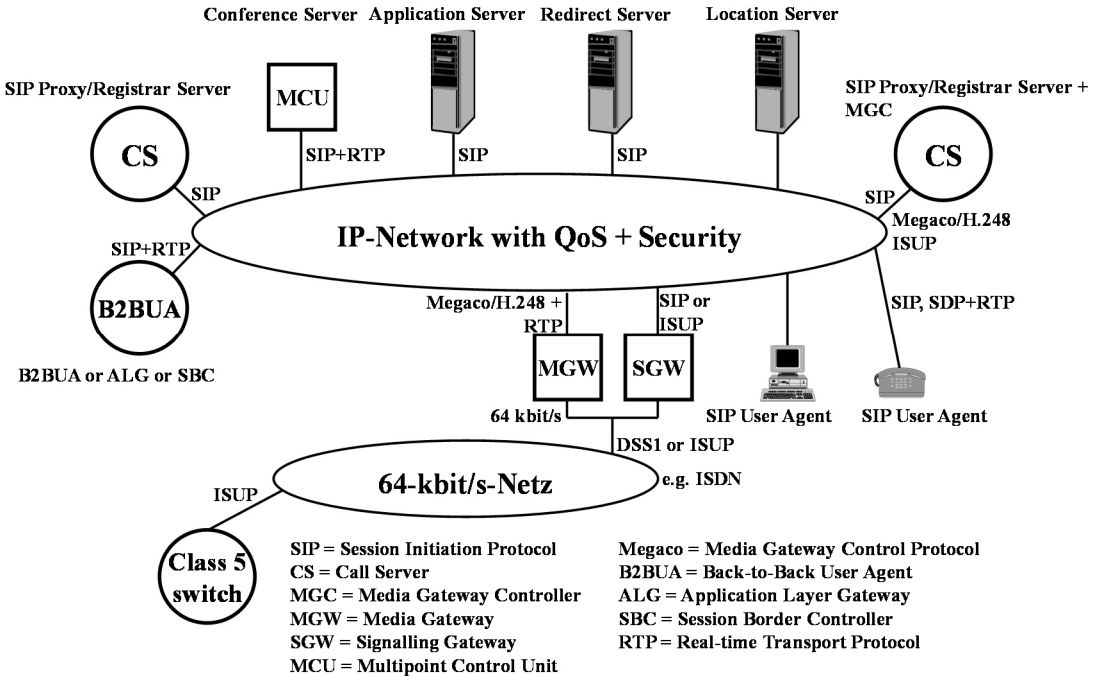


Figure 2.1: Conceptual structure of a SIP-based NGN (Trick and Weber, 2009)

Note that the deflection of both signalling and media streams to certain network elements potentially counteracts the NGN key feature separation of call/service control and media data transport (see section 2.1) but, on the other hand, potentially facilitates other NGN key features such as consideration of obligatory legal requirements. (Weber, 2012)

SIP B2BUA is working as two User Agents sitting with their back to each other. So a User Agent contacting a B2BUA will only face one site of the B2BUA. The other side he will never see. This means a B2BUA can be used for topology hiding. It has the ability to separate two networks from each other. Furthermore a SIP B2BUA is enabled to create SIP requests and responses, which means it has the ability to create SIP dialogs on its own (Zhuang *et al.*, 2013). A B2BUA can also be used for NAPT at the borders of a network. It will furthermore not only terminate SIP, but also the payload (e.g. RTP). One special field of application of a B2BUA feature is in context with

application servers. On the basis of this functionality rather complex value-added services can be realised.

2.3 SIP-based peer-to-peer networks

P2P networks include a number of advantages when compared with centralised networks, from scalability, and no single point of failure to cost reduction and cost-splitting. Such characteristics make P2P an interesting alternative to centralised approaches. This section gives an overview of SIP and peer-to-peer (P2P) networks, starting with an overview of different existing peer-to-peer models, then presenting the standardised SIP peer-to-peer models, together with the meaning of P2P within the RFC 3261.

2.3.1 Overview of different peer-to-peer models

Peer-to-peer systems are inherently scalable and reliable because of the lack of a single point of failure. P2P systems, in the purest form, have no concept of centralised components as shown in Figure 2.2. All participants are peers and communicate in distributed, potentially untrusted environment, to achieve a certain objective such as locating music files or users. Some earlier developed P2P systems, with Napster being a notable example, were partially based on centralised elements. Such P2P systems are called hybrid P2P systems (Singh and Schulzrinne, 2004).

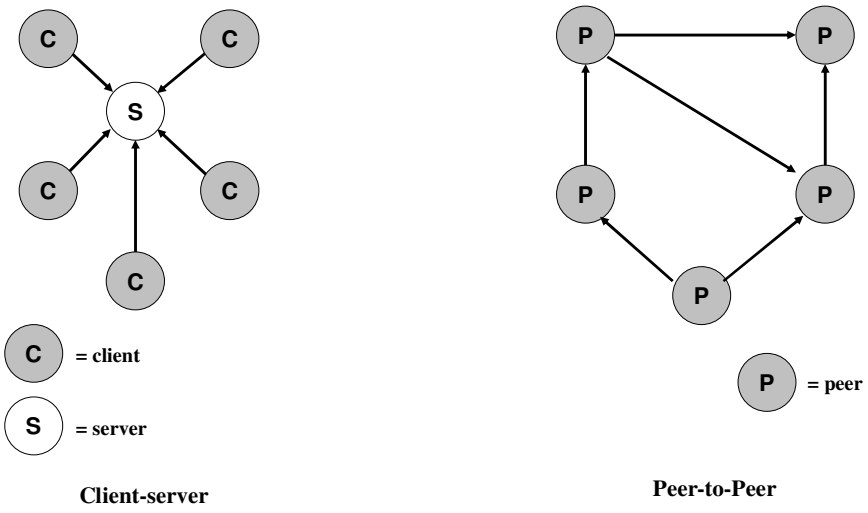


Figure 2.2: Client-server versus peer-to-peer distributed systems (Singh and Schulzrinne, 2004)

Three different models (generations) were proposed by prior research according to (Eisenschmid, 2005) and (Harjula *et al.*, 2004) as illustrated in Figure 2.3. Each new generation is a derivative. The first generation designated as hybrid P2P model is a semi-centric model with at least one centric checkpoint, for instance one centric peer as an index server for available data. Communication can be client-server and peer-to-peer. Representative implementations of the first generation are for example Napster, Skype and SIP according to (IETF RFC 3261, 2002). The second generation, also denoted as pure P2P model, is a fully distributed peer-to-peer architecture, which can come in two different flavours. It can be based on a special topology (structured) or the peers can be organised without any structure. An exemplary implementation of this generation is Gnutella. The third generation called super P2P model is an enhancement of the hybrid P2P model. The centralised checkpoints are substituted by a peer-to-peer network of super nodes. The interaction between super nodes and ordinary peers is based on client-server communication. A peer can become a super node for instance

in dependency of bandwidth and performance. Representative implementations are KaZaa and Skype.

For a better understanding the three models are shown in a general overview (see Figure 2.3).

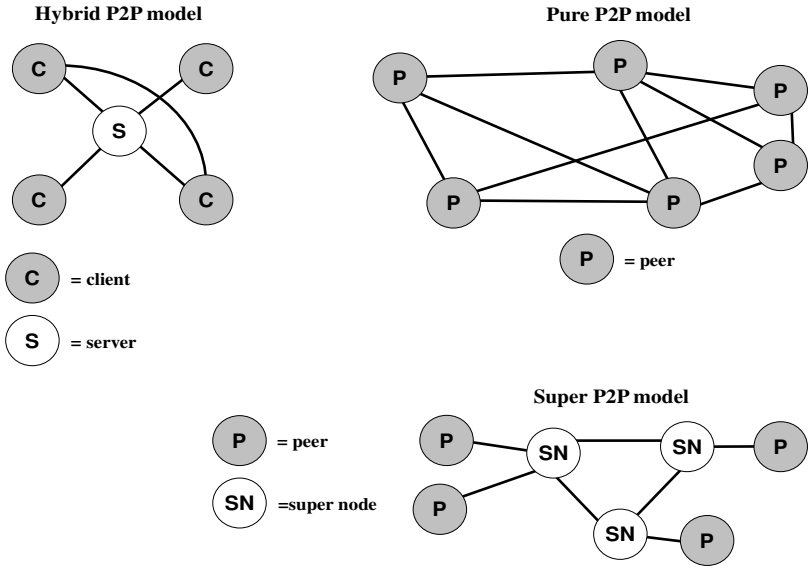


Figure 2.3: General overview of peer-to-peer models

The Pure P2P model can be divided into two different types (Bischof, 2005). The first mentioned are the unstructured P2P networks, which make use of different search algorithms like flooding (forwarding the request to all known neighbours) or random-walk (every node decides randomly to whom the request is forwarded). These algorithms may lead to long delays between request and response. The second types of the Pure P2P model are structured P2P networks. In these networks all peers are organised in a specific topology (e.g. logical ring topology).

Every peer is responsible for an appropriate amount of IDs (Identity) concatenated to other peers in the topology. Structured P2P networks have one disadvantage compared

to unstructured ones; only exact searches are possible (no keywords), but the search algorithm is more efficient. Structured peer-to-peer networks are mostly organised by distributed hash tables (Bravo *et al.*, 2012). Distributed hash tables (Garces-Erice *et al.*, 2004) are similar to hash maps (Cormen *et al.*, 2009), which are organised by key value pairs. Each key is a unique identifier for information that should be delegated to other peers, for example a file.

2.3.2 P2PSIP

First some benefits of P2PSIP are named as follows:

Organisations and service providers can save costs. There is no need to host dedicated servers in data centres with high availability and pay for energy and bandwidth. Traditional service providers may move to a more open end-to-end user application, which is as already stated in section 1.1 a crucial point of this research work. P2PSIP does not depend on service provider for signalling and media path. It can use end-user devices on public Internet. The main benefits of P2PSIP are for end-users. They can provide their own services and of cause make use of services provided by other end-users. (Singh, 2010)

The IETF P2PSIP WG (Working Group) is chartered to develop protocols and mechanisms for the SIP and P2P. The IETF's P2PSIP WG discussed two different methods to combine SIP and P2P. The first one is the so-called SIP-using-P2P method and the second one is called P2P-over-SIP (Singh and Schulzrinne, 2006). In SIP-using-P2P, a P2P overlay is connected to SIP network elements (e.g., SIP User Agent) and replaces the functionality of the Location server. Such an implementation does not

need SIP Proxy or SIP Registrar servers. Figure 2.4 gives an overview of SIP-using-P2P.

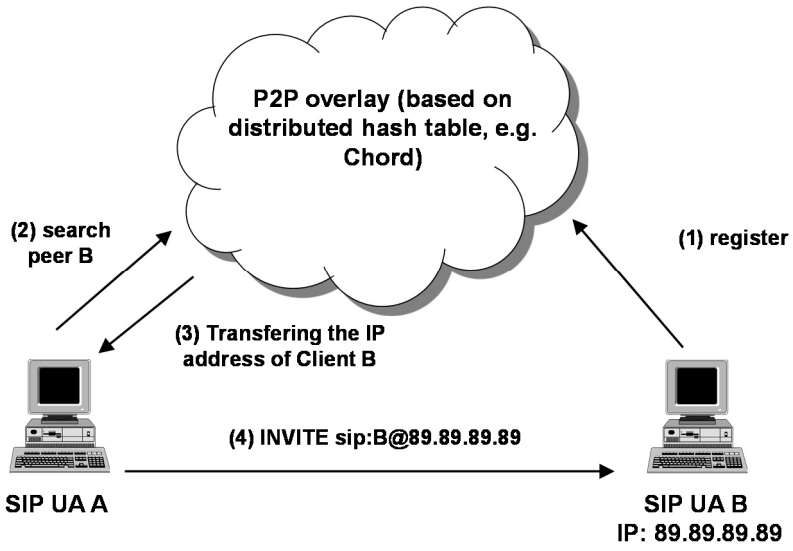


Figure 2.4: SIP-using-P2P (Trick and Weber, 2009)

The SIP-using-P2P concept will define interfaces for using existing peer-to-peer networks to store the location data of each SIP User Agent into the P2P overlay. As illustrated in Figure 2.4 SIP User Agent B uses a method of the implemented P2P algorithm to register at the P2P overlay network (see step 1). Thus the context of the IP address and the subscriber's name is stored as a key-value pair within the P2P overlay, where the IP address is the value and the subscriber's name represents the key. Before subscriber A can send a SIP INVITE request directly to SIP User Agent B he has to request its IP address. Hence subscriber A sends a lookup request with the key that represents the subscriber's name (here: B) to the P2P overlay (see step 2). As a response to the lookup request the subscriber A receives the IP address of client B (see step 3), so the SIP User Agent A can send the SIP INVITE request directly to subscriber B (see step 4).

The advantages of such architecture are that it may reuse an optimised and well-defined external P2P network (Kim and Jeong, 2013), such as Chord. The P2P overlay network is based on an algorithm that supports the storage and retrieval of location information (context of IP address and SIP subscriber's name). Therefore a defined location service interface is defined. Finally, the P2P overlay could also be used by other signalling protocols than SIP.

The P2P-over-SIP concept integrates a peer-to-peer algorithm into the signalling managed by SIP (see Figure 2.5). No changes have to be made for existing SIP components regarding their functionality, because the signalling protocol has not changed.

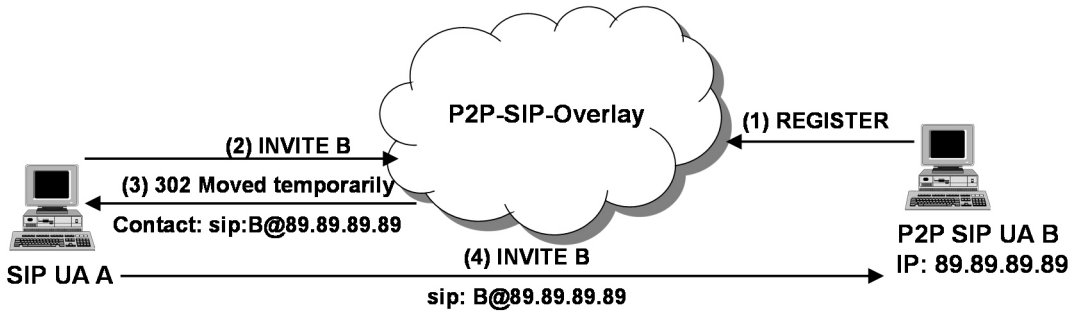


Figure 2.5: P2P-over-SIP (Trick and Weber, 2009)

As depicted in Figure 2.5, the User Agent B registers with a SIP request REGISTER at the P2P SIP overlay network (see step 1). Though the context of the IP address (here: IP 89.89.89.89) and the subscriber's name (here: B) is stored within the P2P SIP overlay network. If subscriber A tries to establish a SIP session with subscriber B, it will send a SIP INVITE request to a peer, which is also a SIP network element, within the P2P SIP overlay (see step 2). The requested peer will answer for example with a response indicating status 302, and containing the temporary SIP URI of subscriber B.

Upon the receipt of the temporary SIP URI the User Agent A sends an INVITE request directly to User Agent B (see step 4).

The advantages of the P2P-over-SIP method are that an arbitrary P2P algorithm over SIP can be implemented into SIP without any changes in the semantics. Because of the integration of the P2P algorithm there is no dependency on external P2P networks. Another benefit is that the reuse and interoperation with existing components, such as voicemail, is supported. Furthermore built-in NAPT/media relays can be used.

One big disadvantage is the emerging message overhead caused by the alter header information of original SIP messages in order to accomplish more functions, like indicating nodes joining or leaving the overlay (Pu, 2006) (Singh and Schulzrinne, 2006). So the P2PSIP WG recommends a SIP-using-P2P based concept, which is discussed in the following.

The IETF's WG on P2PSIP has developed a new approach to make use of any existing P2P algorithm. This approach is the so-called RELOAD (REsource LOcation And Discovery) system (Jennings *et al.,* 2012). RELOAD is a peer-to-peer signalling protocol, which provides its clients with an abstract storage and messaging service between a set of cooperating peers that form the overlay network. RELOAD has been designed to support P2PSIP networks, but it can also be used by other applications. In January 2014 the IETF published the RFC 6940 that defines the Resource Location And Discovery Base Protocol (IETF RFC 6940, 2014).

2.3.3 SIP hybrid peer-to-peer

A SIP-based peer-to-peer communication network (equivalent to the Hybrid P2P model) matches the description given in section 2.2.1. The other P2P models described in section 2.3.1 can also be implemented based on SIP. The main differences between the centralised and distributed approaches are the storage of information (e.g. the matching between temporary and permanent SIP URIs) and the distribution of special functions, for example application servers and call servers can be peers. Figure 2.6 depicts a SIP-based P2P network equivalent to the Hybrid P2P model which is proposed as a new efficient approach by the author of this thesis in (Lehmann *et al.*, 2008a) and (Lehmann *et al.*, 2008b).

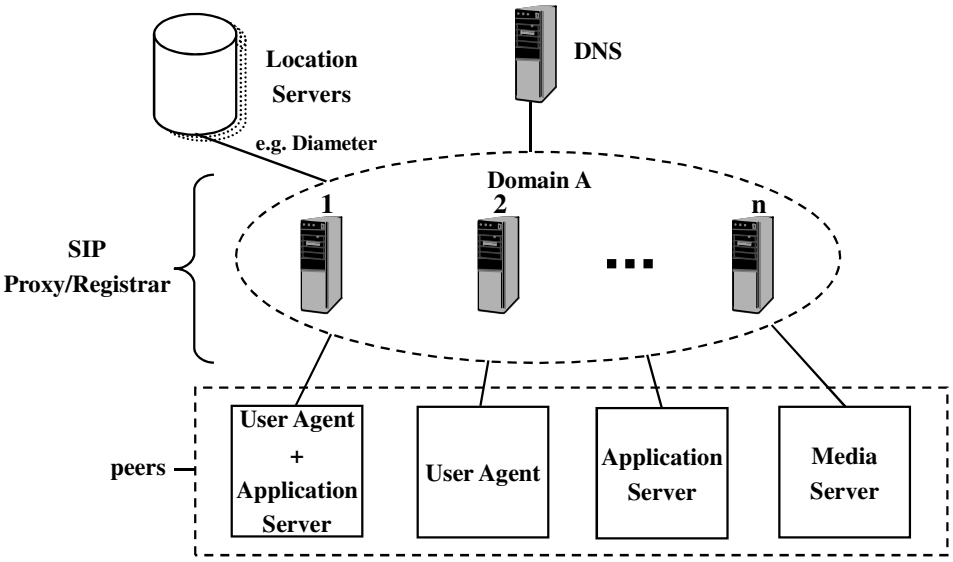


Figure 2.6: New hybrid SIP-based P2P approach

The proposed P2P overlay network is implemented by using multiple SIP Registrar/Proxy servers. The domain for these SIP elements can be resolved by DNS (Domain Name System) (IETF RFC 1034, 1987), (IETF RFC 1035, 1987), or dynamic DNS (IETF RFC 2136, 1997). Further SIP network elements can simply register with

the overlay network by using the mechanism shown in section 2.2. A pool of Location servers is connected to the overlay. The Location server pool is holding the data of the registrations from the peers to provide the routing. Now application servers and media servers can easily be connected to the overlay as peers to provide value-added services. The innovation of this concept is that all SIP elements except SIP Proxy and Registrar servers are connected to the overlay as peers regarding to (Lehmann *et al.*, 2008a), (Lehmann *et al.*, 2008b). This means that also other SIP elements than SIP User Agents will be equipped with the ability to register on the SIP peer-to-peer network. In addition, there is no need to implement extensions to SIP elements other than the core elements, the SIP Proxy and Registrar servers that have to be upgraded with peer-to-peer overlay functionalities. Also, the Location Servers can be implemented into a distributed hash table (DHT) that stores the permanent SIP URIs as keys and the dedicated values contain the temporary SIP URIs. This results in a combination of SIP Proxy/Registrar and Location servers.

2.4 Conclusion

This chapter illustrated the demand for packet-based telecommunication networks, focusing on the concept of NGN, as defined by ITU-T and ETSI TISPAN, and the separation of transport and service stratum.

Section 2.3 introduced SIP as the de-facto standard signalling protocol in NGN, hence SIP has been considered as the protocol of choice for signalling within this research. A SIP-based peer-to-peer infrastructure has been introduced in section 2.3.3, integrated with the general architecture of SIP-based communication as standardised by IETF and presented in section 2.2. Using the SIP-based peer-to-peer infrastructure

the partition of transport and service stratum will be preserved as specified by the concept of NGN. Section 2.3.3 provided a novel approach for SIP elements others than SIP User Agents to register in a peer-to-peer manner. Therefore only the functionality to register has to be implemented inside of these SIP elements, which is a standard feature, and no further extensions regarding SIP and the peer-to-peer overlay must be realised. The approach allows standard SIP elements without any extensions beyond RFC 3261 to be used as peers. Following the description of the fundamental architecture of a SIP-based peer-to-peer NGN, the following section will continue with illustrating further principles regarding services and telecommunication.

3 Services and telecommunication

This chapter expands on the term service within information technology (section 3.1) as well as in the telecommunication context (section 3.2). The understanding of the term service is essential because it is in the focus of this research work. The convergence of telecommunications networks and information technology still takes place (Leon, 2014). One result of this convergence is that approaches such as Service-Oriented Architecture (SOA) also find its way into telecommunication. So the notion of SOA is depicted and the link between SOA and telecommunication is illustrated (section 3.1). After that the concepts for service provisioning in NGN/IMS are shown (section 3.3) and in addition standardised procedures for service composition are discussed (section 3.4) to define the requirements regarding to services and their composition for the research framework.

3.1 Service-Oriented Architecture (SOA)

The Organization for the Advancement of Structured Information Standards (OASIS), which is a non-profit consortium that produces worldwide standards for SOA see (OASIS Standard, 2006).

In (Rosen *et al.*, 2008) SOA has been defined as: SOA is an architectural style for building enterprise solutions based on services. More specifically, SOA is concerned with the independent construction of business-aligned services that can be combined into meaningful, higher-level business processes and solutions within the context of enterprise. A Service Oriented Architecture is a software architecture of services,

policies, practices and frameworks in which components can be reused in order to achieve shared and new functionality (ITU-T M.3060/Y2401, 2006). The architecture involves loosely coupled, location independent services generally using so-called "find-bind-execute" paradigm for communication. Any given service may assume a client or a server role with respect to another service, depending on situation. An essential characteristic of a SOA is that it provides published contract-based, platform and technology neutral Service Interfaces. This means that the interface of a service is independent of its implementation. In practice, interfaces are defined using ubiquitous IT standards such as XML (Extensible Markup Language), HTTP (Hypertext Transfer Protocol), SOAP (Simple Object Access Protocol), and WSDL (Web Services Description Language). (ETSI TS 188 001, 2006)

The main features of SOA can be summed up as follows, regarding to (Erl, 2007) and (Rosen *et al.*, 2008):

- loosely coupled services: Coupling refers to the extent of dependency between modules, components, or service consumers and providers.
- location independent services: Services are designed to be location-transparent, they are accessible to any authorised user, from any location.
- reusable services: Services are shared and reused as building blocks in the construction of processes or composite services.
- modularity and granularity: Services themselves can be composed from other modular services, and can be mixed and matched as needed to create new composite services.
- autonomy: An autonomous service's life cycle is independent of other services.

- service discoverability: Services are supplemented with communicative meta data by which they can be effectively discovered and interpreted.
- stateless services: Services are not depending on the context or state of other services – only on their functionality.
- any service: any given service may assume a client or a server role with respect to another service, depending on situation. (EDIN 0531-1652, 2006)
- published contract-based, platform and technology neutral service interfaces: This means that the interface of a service is independent of its implementation. (EDIN 0531-1652, 2006)
- composable: Services can be composed from other services and, in turn, can be combined with other services to compose new services.

The term service in SOA's context has to be clarified. The focus of a Service Oriented Architecture is on the functional infrastructure and its business services, not the technical infrastructure and its technical services. An important feature of software architectures such as SOA is that it breaks down the overall structure of a software system into smaller components, which are intended to be flexible building blocks (Krafzig *et al.*, 2005). These software components are exposed pieces of functionalities with the following properties (Hewitt, 2009; Hashimi, 2003):

- A service is defined by an interface that may be platform-independent.
- A service is available across networks.
- A service can be dynamically located and invoked.
- A service interface and its implementation can be decorated with extensions that come into effect at runtime (Josuttis, 2007).

- A service is self-contained (Josuttis, 2007).
- A service hides technical details and allows business people to deal with it. (Josuttis, 2007)
- A service is described within a service description, which represents the information needed in order to use a service. (OASIS Standard, 2006)

The design patterns for SOA follow the already mentioned principles. Furthermore to support some of the principles the so-called "find, bind, execute" paradigm has been defined for SOA according to (ITU-T M.3060/Y2401, 2006) (see Figure 3.1). The so-called "find, bind and execute" paradigm and the SOA principals are main drivers for service composition in telecommunications.

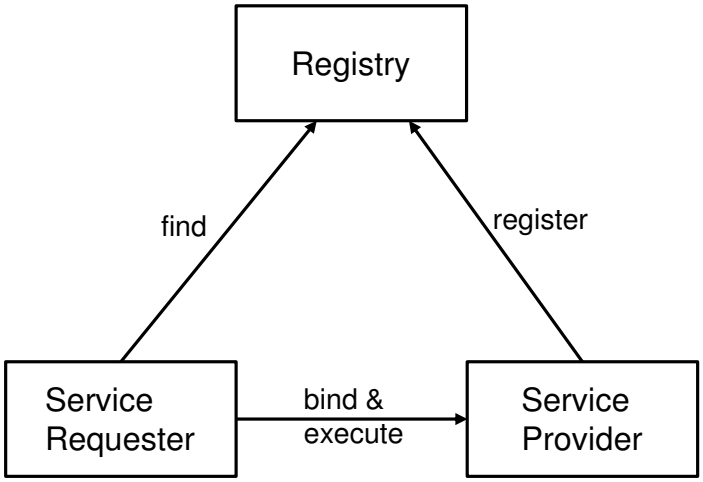


Figure 3.1: "Find, bind and execute" paradigm in SOA (according to ITU-T M.3060/Y2401, 2006)

3.1.1 SOA and telecommunications

At present, the telecom operational enterprises are faced with the problems of how to integrate the existent Operational Support Systems (OSS) to meet the clients' demands quickly and how to provide end-to-end business services with fewer costs (Weifeng *et*

al., 2007). Both (ETSI TS 188 001, 2006) and (ITU-T M.3060/Y2401, 2006) address SOA to be the architectural framework for NGN management architecture. Also in the area of service delivery platforms, SOA is very important (Pethuru, 2014). The Open Mobile Alliance (OMA) has already looked into the lack of standardisation of IMS applications; inspired by the Parlay Group's work, OMA developed the OMA Service Environment, which allows creation of applications that are aligned to SOA principles (Magedanz *et al.*, 2007). Figure 3.2 illustrates the basic architecture of a service delivery platform and its SOA-related functions.

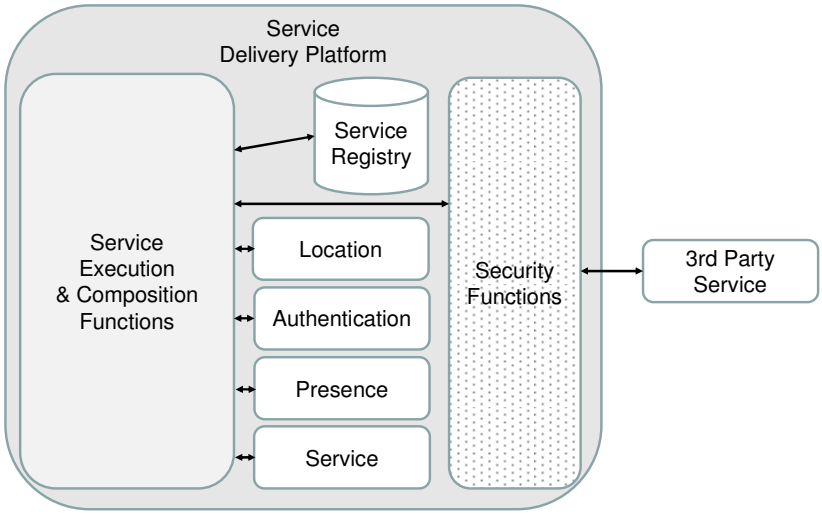


Figure 3.2: Basic Service Delivery Platform architecture

The service execution and composition functions can query and discover services within the service delivery platform and services through secure interfaces provided by the security functions. The service registry holds the service descriptions of both the services within and outside of the service delivery platform. Also compositions of existing services can be managed by the service delivery platform composition functions. The basic functions presented here are integral parts of today's service delivery platforms in telecommunications, e.g. OMA Service Environment (OMA

RRD OSE, 2009).Many SOA implementations are based on web service technologies (BrightStar, 2014) especially standardised APIs (Application Programming Interface) by OMA (OMA ERD OWSER, 2006) and Parlay X (Parlay, 2004). These APIs enable software developers to exploit the capabilities of the underlying network such as IMS/NGN. The design model for web services has changed from the old fashioned SOAP- and WSDL-based interfaces to the so-called RESTful web services, because REST (Representation State Transfer) simplifies the use of services. REST defines a set of architectural principles by which web services can be designed that focus on a system's resources, including how resources states are addressed and transferred over HTTP by a wide range of clients written in different languages (Rodriguez, 2008).

The first company that published an API was British Telecom in 2007. Other European companies followed such as Orange, Deutsche Telekom, and Telefonica. All these products are based on web services (Branca and Atzori, 2012).

3.1.2 Service Delivery Platform in NGN

In the NGN environment, network providers will make use of a service delivery platform with respect to the provisioning of value-added services. By this platform services can be developed, provided and be integrated into other value-added services. The service delivery platform is located in the application layer and connected to the service and transport layer through abstract interfaces. A simplified access to the network functionalities is provided by the abstract interfaces, e.g. location or authentication. Beside the offered services a service delivery platform also supports the development of new services and provides standardised interfaces to the Service

Creation Environment (SCE), AAA, OSS and Business Support System (BSS) (Lehmann and Trick, 2007).

According to the Moriana Group (Moriana, 2013) a service delivery platform is defined as follows: “The term Service Delivery Platform refers to a system architecture or environment that enables the efficient creation, deployment, execution, orchestration and management of one or more classes of services. As such, the service delivery platform is the key component of the telecom Service Layer.”

Initially, the term service delivery platform was characterised to describe a common service architecture specifically designed to deliver mobile content and messaging services. Between the years 2003 and 2006, further development of the service delivery platform integrated the support for typical telecommunication services like voice, multimedia, location, presence and charging services. This second generation, service delivery platform 1.0, fully embraced standard IT technologies and offered a secure and managed third party access to network services through telecom web services standards, such as Parlay X. service delivery platform 2.0, which is the latest, third generation of service delivery platforms, is constructed regarding to the SOA principles, which enables service integration, orchestration and lifecycle management. Furthermore, the third generation service delivery platform facilitates service migration from legacy networks (e.g. PSTN (Public Switched Telephone Network)) to IP-based networks and the standard IMS architecture (Moriana, 2013).

In Figure 3.3 the entire architecture of a today’s service delivery platform is illustrated.

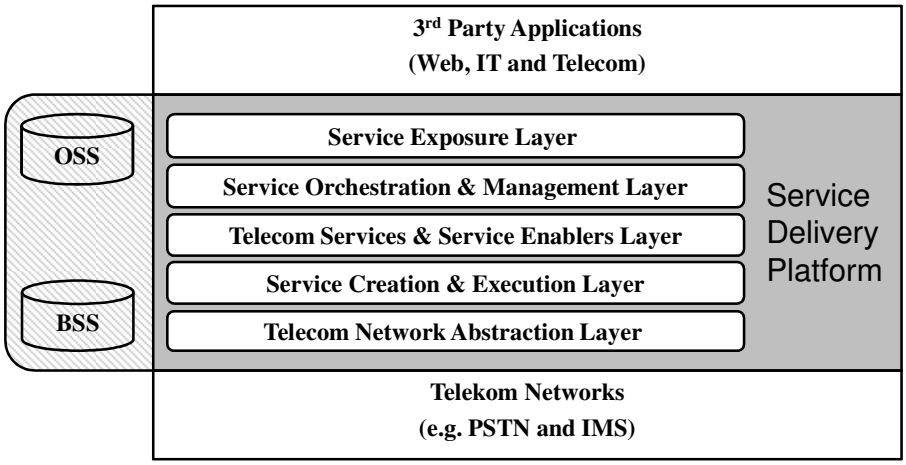


Figure 3.3: Service Delivery Platform architecture (Moriana, 2013)

According to Moriana Group the service delivery platform 2.0 architecture consists of the following layers:

- Service Exposure Layer (SEL): The SEL creates through standardised and secure interfaces, for instance Parlay X, access to voice and data services (e.g. SMS, MMS (Multimedia Messaging Service) or user location), so that other service providers and companies can make use of them. The complexity is hidden by abstraction of the interfaces; allowing even non-telecommunications developers to use the offered capabilities (Otto, 2005).
- Service Orchestration and Management Layer: This layer implements the principles of SOA, which can be used to integrate with OSS and BSS (Lu *et al.*, 2008).
- Telecom Services and Service Enablers Layer: This layer provides deployable telecom services and their enablers to process them. The enablers are highly abstracted interfaces which allow services to make use of telecommunication resources (Lu *et al.*, 2008) and (Lehmann and Trick, 2007).

- Service Creation and Execution Layer: This layer supports the creation of services out of a set of predefined and already existing services, as well as new services. Also the service execution is part of this layer. The Service Creation and Execution Layer integrate functionalities to develop, activate, deactivate and configure services (Lehmann and Trick, 2007).
- Telecom Network Abstraction Layer (NAL): The NAL delivers standardised interfaces to most important network elements and services, for example user location and status, accounting, call management, multimedia services and messaging (SMS, MMS, Instant Messaging and email). The abstraction layer hides peculiarities of diverse underlying networks and manufacturer-specific implementations of net services. For instance, applications running within a service delivery platform should have the ability to request the users location, whether this information is coming from a mobile phone network, fixed-line or VoIP network (e.g. to provide emergency calls). Thereby the applications, which are running on the service delivery platform, will become portable and independent of the particular network. To provide this abstraction standards like SIP Servlets, OSA/Parlay (Open Service Access) JAIN (Java APIs for Integrated Networks) SLEE (Service Logic Execution Environment) or Parlay X will be used (Lehmann and Trick, 2007) and (Lu *et al.*, 2008).

3.2 The concept of service in the telecommunications context

The telecommunications (communications engineering) area deals with capturing, processing, transmitting and storing information (communications). For this purpose,

telephony companies provide services (telecommunication services). The definitions of services are often misunderstood because of the utilisation of different terminologies describing the same, such as in (Kühn, 1991), (TINA-C, 1997), (ETSI TS 122.228, 2009), (ETSI TS 122.105, 2008), (ETSI TS 122.101, 2009), (ITU-T Recommendation I.211, 1993). This demonstrates that there are different meanings of services given by the standardisation bodies, which may result in erroneous understanding and usage of these terms.

Traditionally, telecommunication systems and protocols are based on layered structures, like the Open Systems Interconnection reference model. Functions that are required for communication are modelled in overlying layers, where each layer fulfils a specific task, such as encoding and decoding of data. An interface offered from one layer to an upper layer to make use of its service(s) is called Service Access Point (SAP) (see Figure 3.4).

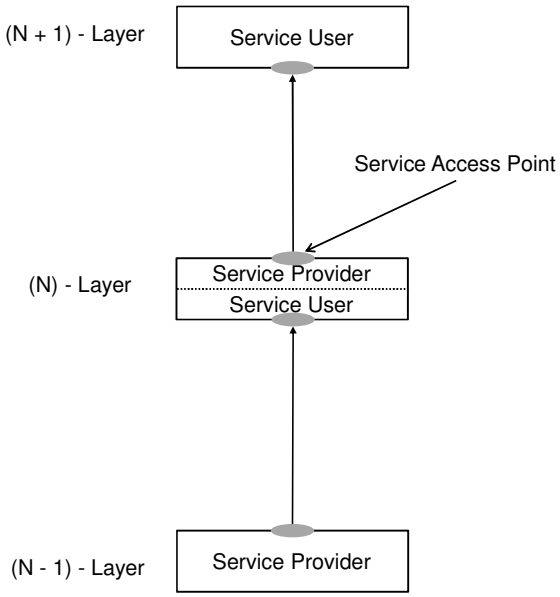


Figure 3.4: Service Access Point

The service user accesses the service of the lower layer through the SAP. Regarding to the SAP model, the telecommunication systems are following a common functionally layered reference model (the OSI-Reference model (Open Systems Interconnection)) with a clear concept of service orientation: clearly defined service interfaces with access points, clearly defined roles of service provider and service user, reusable components and loosely coupling.

Services are basically divided into bearer services, teleservices and supplementary services. A telecommunication service is a combination out of a basic bearer service without or with supplementary service(s) and a basic teleservice without or with supplementary service(s) as shown in Figure 3.5 referred to (ETSI TS 122.001, 2009) and (ITU-T Recommendation I.210, 1993).

<table>
<tr><td colspan="2">telecommunication services</td></tr>
<tr><td rowspan="2">teleservice</td><td>basic teleservice +
supplementary service (s)</td></tr>
<tr><td>basic teleservice</td></tr>
<tr><td rowspan="2">bearer service</td><td>basic bearer service +
supplementary service (s)</td></tr>
<tr><td>basic bearer service</td></tr>
</table>

Figure 3.5: Categorisation of telecommunication services

In addition to the already described service definitions a further service type, value-added services, is existing. The term value-added service simply states that there is a bearer service and teleservice service or simply a teleservice (without or with supplementary service(s)) that is enhanced by additional application logic to add more value to the service. The following paragraph will define the term value-added services to clarify the meaning of services in this research work. Thus all following sections

will refer to the definition of the term value-added service, illustrated by the following paragraph, whenever the notion service will be used.

“Value-added services (VAS) are functional properties which will offer certain comfort to consumers. Consumers will recognise additional benefit by value-added services. They are based on a combination of one or more bearer services (here solely IP bearer services), and one or more teleservices, and optionally one or more supplementary services. They also provide benefits, i.e. added value, in respect to the original (core) services, e.g. a normal telephony service, cannot. VAS can be an add-on to basic teleservices and they can sometimes stand-alone (e.g. non-call related services). VAS also have a certain time dimension associated with them. A value-added service today can become a basic service in the future when it becomes sufficiently common place and widely deployed, and for example, is no longer used as a differentiation feature among operators.” (MobileIN, 2004) Furthermore value-added services are provided by application servers and media servers.

The three named above service categories bearer service, teleservice and value-added service can be allocated to different strata in NGN. The transport stratum provides the bearer service. Teleservices and the most supplementary services are realised through SIP and the call servers in the service stratum. Application servers and media servers provide the value-added services. The separation of the service logic from the routing entities (call server) results in lesser dependencies between the network and services. Introducing new services will be much easier following the concept of NGN.

3.3 Service provisioning in IMS

Currently, IMS service provisioning is not automated to support service aggregation. The service stratum must have knowledge about all existing services in the application stratum, which may consist of multiple application servers. Service invocation via SIP is always routed by a call server, in IMS known as CSCF (Call Session Control Function).

So some configuration and management has to be done to provide new services. For this purpose the call server is connected to the AAA server, which is called Home Subscriber Server (HSS) in IMS. The HSS represents the central element of the IMS and is a master database that supports the IMS network entities, containing the subscriber profiles, performs authentication and authorisation. This server holds all information related to subscribed services. A call server, the S-CSCF (Serving-Call Session Control Function), routes SIP requests to an adequate application server depending on the information, named Initial Filter Criteria (IFC), which was requested from the HSS. Depending on the IFC the call server decides which application server is the appropriate one. An IFC contains one or multiple criteria, so-called Service Trigger Points, each describing conditions that should be checked to discover whether the indicated AS should be contacted.

Hence an IFC consists of one or more Service Trigger Points and entries of application servers to contact if the trigger rule matches. Multiple IFCs can be integrated under a User Profile that is a collection of user-specific information, which is permanently stored in HSS.

This concept lacks of handling SIP responses or subsequent requests, so only initial requests can be forwarded to application servers. In some cases it might be useful to forward special formed responses to application servers, e.g. if a transcoding service is needed due to non-matching codecs within the SIP body containing the session description.

Summarising, the service provisioning in IMS contains three steps (Poikselkä and Mayer, 2009). Define possible service or service sets, create user-specific service data in the format of initial filter criteria, and pass an incoming initial request to an AS. These three steps have to be accomplished by the OSS.

3.4 Service composition concepts

Different concepts for service composition can be identified with regard to NGN/IMS. Those concepts will be presented in the following sections.

Conceptually, the meaning of service composition has to be defined. In (ITU-T Y.2234, 2008) the term service composition is defined as follows:

“Service composition is the capability of creating new services from other existing services. Service composition can occur in a static or in a dynamic way. While in static composition, composite services are defined in advance, dynamic composition sends the request for service discovery using the service description to find the needed services and composes the service during run time”.

In IMS, the S-CSCF has implemented functionality to support simple service compositions based on the IFCs (see section 3.3). Therefore a sequence of filter criteria

is defined inside of a service profile. This is a static composition regarding to (ITU-T Y.2234, 2008).

Service composition and service invocation are closely linked together, as a service composition will not work without service invocation. During the service composition, the particular services, which will form the resulting composition, are invoked. Summarising, the service composition is formed by invoking existing services to create a new service, which called a composite service.

3.4.1 Service Broker in IMS

The "Study on Architecture Impacts of Service Brokering", discusses the concept of the SCIM as part of an AS and Service Broker as part of Open Service Access-Service Capability Server (OSA-SCS) (3GPP TR 23.810, 2008). The so-called service capabilities are modular building blocks that can be reused across different application servers (ETSI DTR 01024, 2005). A special SIP application server called SCIM to control the interactions between services has been proposed in (3GPP TS 23.218, 2006). But the specification does not specify the functionalities of SCIM, i.e. the way it should coordinate multiple invocations of the service capabilities and the way that it should handle the incompatibilities between service capability invocations (Gouya *et al.*, 2006).

In (Gouya *et al.*, 2006) the SCIM is used to invoke services (Service Capabilities) and interact with other SCIMs may it be different SCIMs of a domain, different domains and/or different service providers. A possible implementation method was specified within (JSR 289, 2008). The principles for application composition and autonomy were adapted from the Distributed Feature Composition (DFC) architecture (Jackson

and Zave, 1998) (see Figure 3.6). Within the SIP Servlet specification an application router has been defined, which controls the sequential invocation of SIP applications. These SIP applications then represent service capabilities.

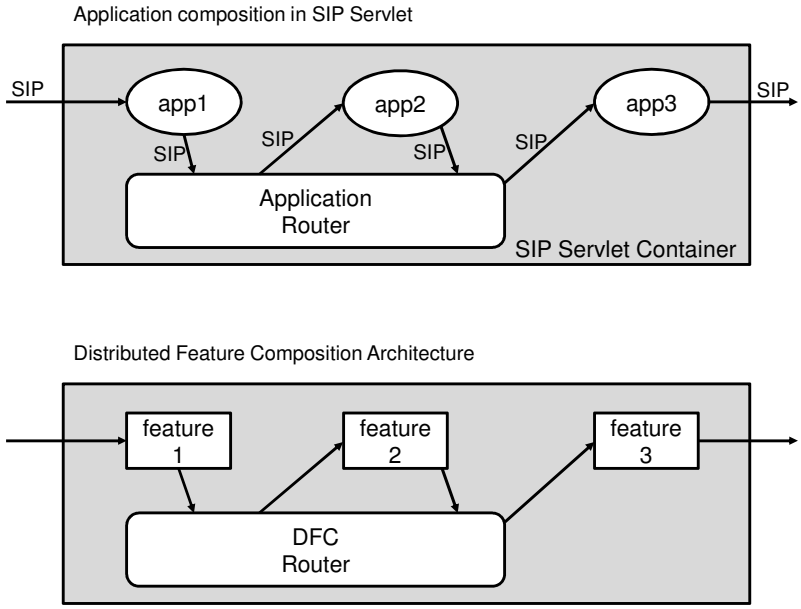


Figure 3.6: Application composition in SIP servlet and DFC (according to Cheung and Purdy, 2008)

The MultiService Forum (MSF) for IMS specified another possible implementation. Within (MSF IA SIP 005, 2004) and (MSF IA SIP 006, 2005) based on the SIP Route header (IETF RFC 3261, 2002) so-called service chains will be realised. A service chain is a sequential invocation of services. A so-called Service Broker (see Figure 3.7) representing a SCIM organises the service identification and routing. One difference, regarding the SCIM as defined in SIP Servlets, is the possibility to set a SIP Route header so that direct routing between application servers can be initiated by the Service Broker. As already mentioned earlier in this section, a Service Broker is defined as part of OSA-SCS. The OSA defines an architecture that enables service application developers to make use of network functionality through open standardised

interface, i.e. the OSA APIs and Parlay X web services (ETSI TS 123.198, 2010). The SCS is defined as a functional entity providing OSA interfaces towards an application. These applications are not part of the.

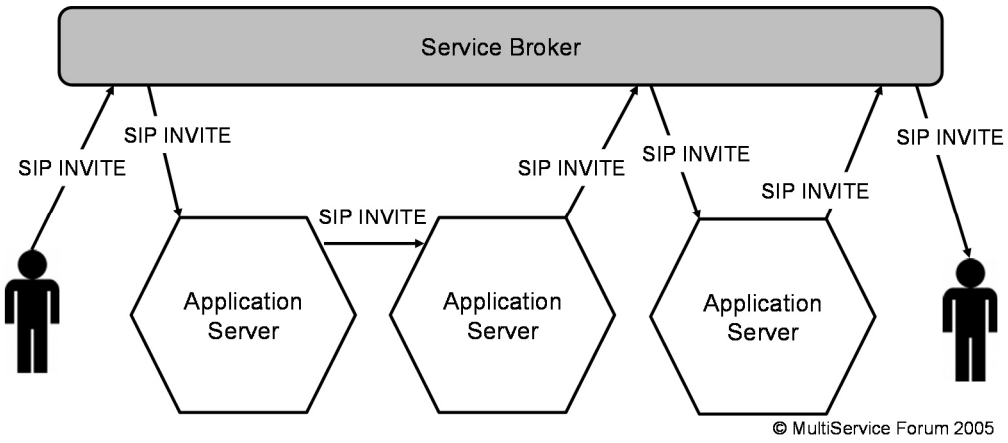


Figure 3.7: Application composition with Service Broker (MSF IA SIP 006, 2005)

In (ETSI TS 122.127, 2010), the Service Broker is defined as follows: "[The] Service Broker function shall enable the delivery of multiple services in an operator network in a managed and controlled fashion. Therefore whenever an event occurs, there is a need to ensure that the set of applications or services that may act upon that event are invoked in a manner that does not conflict with any other application or service defined in the provisioned package of applications or services."

This means that a Service Broker is able to combine different applications and service capabilities. The service compositions can be static, if they are predefined and for instance stored as service profiles, or they can be dynamic. It is not defined how dynamic compositions should be created. Generally it can be said that a Service Broker is a specialised application server within a centralised approach for service composition.

3.4.2 Service composition and web services

The ITU-T has recommended open service environment (OSE) capabilities in (ITU-T Y.2234, 2008), built on NGN with the aim to enable enhanced, flexible service creation and provisioning. To allow for OSE functionality, standard interfaces are necessary to ensure service reusability, portability, and accessibility by NGN application providers and developers. The OSE is composed of different functional components (ITU-T Y.2234, 2008).

- Service Management – supports service tracking, logging and utilisation for instance.
- Policy Enforcement – e.g. protects services from unauthorised requests.
- Interworking with Service Creation Environment – supports interworking functionalities for open service creation environments (e.g. OSA/Parlay, Parlay X, OMA).
- Service Development Support – supports amongst others re-use of services.
- Service Coordination – provides coordination of services and applications with capabilities.
- Service Discovery – supports the discovery of a service based on different criteria (e.g. location, classification). An example of discovering criteria is implemented in the universal description, discovery and integration (UDDI) specification of web services framework.
- Service Registration – provides the registration of services, including configuration, activation, publication, and deregistration.

- Service Composition – supports the composition of services statically and dynamically. Also a composition language is provided that describes the interaction among services.

The use cases for OSE within (ITU-T Y.2234, 2008) are focused on web services technologies. By utilising web services combination of existing services can be established.

Parlay X and OMA (Open Mobile Alliance) have to be mentioned with respect to OSE. Parlay X is based on web services to realise an open access to NGN capabilities like location or presence. OMA has specified an own service environment called OMA Service Environment (OMA RRD OSE, 2009), which supports to provide web services by integrating the OMA Web Services Enabler (OWSER) (OMA ERD OWSER, 2006).

Several related studies focused on service composition for NGN based on web services, e.g. (Blum *et al.*, 2009) and (Wu, 2009). The service composition based on web services is very popular within the research sector; also telecommunication companies have already released open APIs exposing telecommunications specific core network functionalities to 3rd party service developers, e.g. British Telecom's BT Web21C SDK (BT, 2008), the Orange Partner program (Orange, 2014) and Deutsche Telekom's Open Developer Portal (Deutsche Telekom, 2014). To put this in context, the offered solutions are aimed at service developers and not for end-users, which means that they are mostly aimed at development companies.

3.4.3 NGSON (Next Generation Service Overlay Networks)

(IEEE Std 1903-2011, 2011) defined a complete functional architecture of IP-based overlay networks that support context-aware, dynamically adaptive, and self-organised networking capabilities, named Next Generation Service Overlay Networks (NGSON). This architecture is also independent of underlying networks, so it fits on top of IMS/NGN as well as P2P.

Huawei proposed the idea of NGSON to IEEE Standards Association Corporate Advisory Group in February 2007 (Townsend, 2009) then, in March 2008, the NGSON was proposed and launched by IEEE Communications Society (Makaya *et al.*, 2011). A first draft of the proposed standard has been published in May 2008 (IEEE P1903/D1, 2008), which included an overview of the NGSON framework. The NGSON helps the providers to organise and improve their business by offering rich services to their users through composing services dynamically and, further, it also helps to provide richer services to the end users by collaboration with other providers. End users will benefit by use of NGSON to create and request services of their choices. The following list gives an overview of important characteristics concerning services and NGSON (IEEE P1903/D1, 2008).

- Standardised access to services offered by different networks
- Service blending across networks
- Provide automated service delivery capabilities
- Dynamic and real-time composition of service components to deliver customer-desired services

The NGSON is another paradigm having context-aware, dynamically adaptive, and self-organising networking capabilities dealing with services from consumer's perspective (IEEE P1903/D1, 2008).

The architecture of the NGSON contains three different entities, namely transport related functions, operation and management functions and service related functions (Lee and Kang, 2012). Through so-called external reference points end users, other NGSONs, underlying networks (e.g. IMS) and services can be tied to the NGSON. Within the NGSON framework different functional entities (FE) and their relationships are defined (see Figure 3.8).

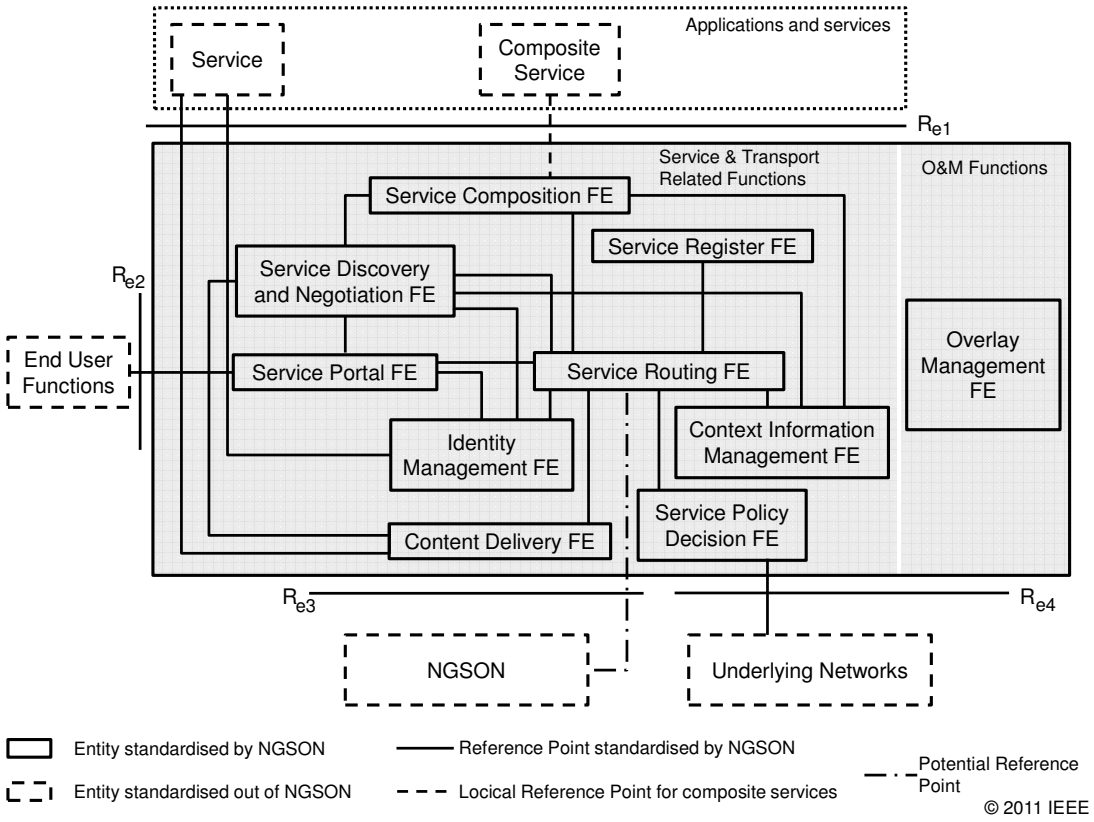


Figure 3.8: NGSON reference diagram (IEEE Std 1903-2011, 2011)

The main functional entities for service composition are the service routing (SR) FE (as central entity to provide routing mechanisms for interaction between services and

end users), service register FE (provides needed information of services e.g. service address), service discovery and negotiation (SDN) FE (supports a service matching mechanism to find services according to given service criteria and a mechanism to find the most suitable service), and service composition (SC) FE (is responsible for execution of composite service requests).

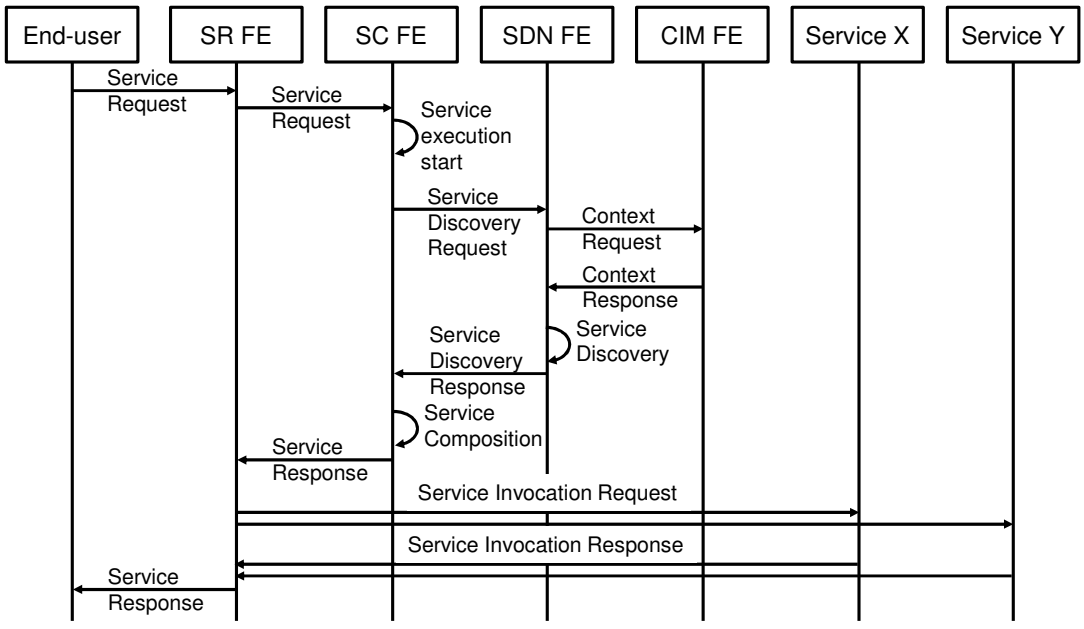
The SC FE supports static and dynamic service composition. Further capabilities are provided by the service composition FE:

- Choice of type of composition (static or dynamic)
- Choice among existing services
- Choice of how to manage the process of composition
- Optimise the composition sequence

The (IEEE Std 1903-2011, 2011) specification only defines the architecture for NGSON and does not include any conceptual recommendations regarding implementation of the FEs. In the annex of the (IEEE Std 1903-2011, 2011), use cases have been specified to illustrate the interworking of the FEs. The following MSC (Message Sequence Chart) shows a service composition based on NGSON functional architecture (see Figure 3.9). A service composition process, as represented by the MSC, can briefly described through the following stages:

1. An End-user sends a service request to a SR FE.
2. The SR FE forwards the service request to the SC FE.
3. SC FE starts service execution.
4. SC FE requests service discovery to SDN FE to find appropriate services for the user's service request.

5. SDN FE requests end-user's context (e.g. location and device type) to the context information management (CIM) FE.
6. CIM FE interprets the request and returns requested context to SDN FE.
7. SDN FE discovers the appropriate services X and Y according to the context information.
8. SC FE receives service discovery response from SDN FE.
9. SC FE performs service composition until service execution ends.
10. SR FE receives service response from the SC FE.
11. SR FE sends requests to invoke the services.
12. SR FE receives the responses for the service invocation.
13. End-user receives the result of the service invocation.



CIM – Context Information Management
SC – Service Composition
SDN – Service Discovery and Negotiation
SR – Service Routing

Figure 3.9: Service composition in NGSON (IEEE Std 1903-2011, 2011)

In general, all of the previous mentioned FEs can be distributed, so in a peer-to-peer network the nodes might consist of NGSON FEs (see Figure 3.10).

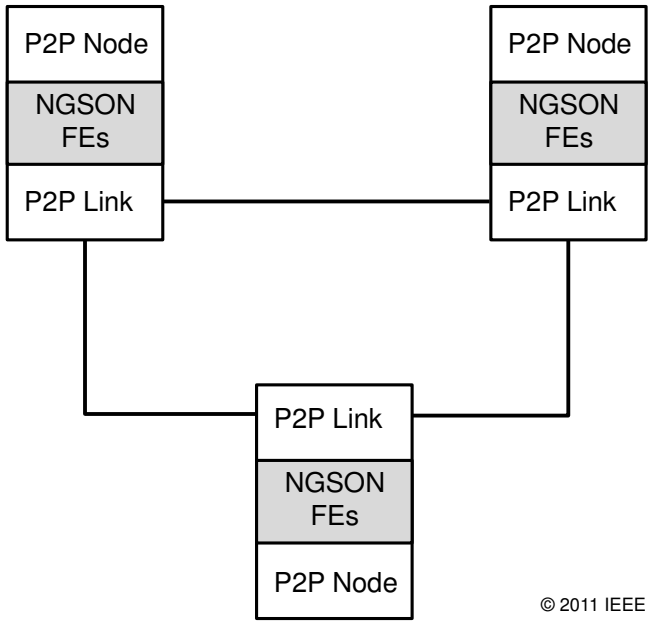


Figure 3.10: NGSON self-organised in P2P (IEEE Std 1903-2011, 2011)

In NGSON, different components communicate and interact in order to share information, to setup a session, deliver content, compose services, etc. For this reason, a communication protocol is required for components interaction (Makaya *et al.*, 2011). In December 2011 IEEE confirmed that three additional standards will be specified to define protocols to be used within NGSON. In (IEEE P1903.1, 2014) protocols among Content Delivery Function Entity (FE), Service Routing FE, Service Policy Decision FE, Service Discovery and Negotiation FE, and Context Information Management FE will be specified to support advanced content delivery capability. Protocols among Service Composition FE, Service Discovery and Negotiation FE, Context Information Management FE, Service Routing FE and Service Policy Decision FE will be specified in (IEEE P1903.2, 2014) to support service composition capabilities. (IEEE P1903.3, 2014) will specify protocols between the Overlay

Management FE and all other NGSON FEs, or NGSON nodes to enable Overlay Management FE involved self-organising management capability. Up to now none of these standards have been published.

3.4.4 Comparison of the existing service composition concepts

The previously mentioned service composition concepts are compared with each other to illustrate their relative strengths and weaknesses. For this purpose different criteria are used on which this thesis is based: principles of SOA, distributed or centralised network architecture, service composition and NGN. A selection of criteria for services identified for SOA in section 3.1 is used as well as the named criteria for NGN in section 2.1. Under the topic service composition four different criteria are used: the type of composition (static or dynamic), user centric services/compositions (combinable by users), a simplified architecture and interworking of the required features to facilitate the service composition and reduction of signalling to minimise signalling overhead. QoS, security, network access and regulatory aspects are not included, because they are not within the research scope of this work. The network architecture is considered under the aspects whether the network is distributed or centralised.

The following Table 3.1 employs a three-level scale, using (-) for missing functionality or feature, (o) for elementary support of functionality or feature and (+) to represent advanced implementation of the functionality or feature. Based on Table 3.1, the NGSON concept satisfies most criteria. One disadvantage of that concept is that so far no protocols are specified for usage in NGSON. Another issue is that NGSON is based on a network on top of an already existing IP-based infrastructure utilising SIP with

respect to this research work. But there is generally no necessity at all for setting up a separated service network. Also the requirement that the service invocation is handled by special functional elements from the network is not essential for a distributed approach. NGSON is more or less a more detailed specification of a service delivery platform for NGN (see section 3.1.2).

Table 3.1: Comparison of service composition concepts

Concept	SOA Principles				Network Arch-itecture		Service Composition					NGN					IP-based
	loose coupled services	reusable services	service discoverability	services description interface	centralised	distributed	Type of composition: static (predefined)	Type of composition: dynamic	user centric	simplified architecture and interworking	reduction of signalling	applicability for arbitrary services	separation of service control and media transport	application server support	support for multimedia services	openness for new services	based on standardised protocol(s)
IMS (IFC)	+	+	-	o	+	-	+	-	-	o	o	o	+	+	+	-	+ (SIP)
SCIM	+	+	-	o	+	o	+	o	-	+	+	o	+	+	+	-	+ (SIP)
MSF Broker	+	+	-	-	+	o	+	+	-	+	o	o	+	+	+	-	+ (SIP)
OSA	+	+	+	o	+	-	+	+	o	-	-	o	o	+	+	o	+
OSE (Web Services)	+	+	+	+	+	o	+	+	o	-	-	+	+	+	+	o	+
OMA/ Parlay X	+	+	+	+	+	o	+	+	o	-	-	+	o	+	+	o	+
NGSON	+	+	+	+	+	+	+	+	+	-	-	+	o	+	+	+	-

3.4.5 Requirements for a new optimised concept in SIP peer-to-peer network

This section introduces the requirements for a new optimised concept based on a SIP peer-to-peer network fulfilling the criteria mentioned in section 3.4.4 and summarising all the requirements derived from the results of the sections within chapter 3.

Conceptually, the new approach must provide dynamic service composition triggered by end users. Fundamental requirements based on NGN characteristics regarding to services must be fulfilled, the applicability for arbitrary services, the separation of service control and media transport, the support for application servers and multimedia services as well as openness for new services. Also the issues reusability, discoverability, loose coupling and composability of services according to the SOA principles must be achieved. The whole framework must be based on IP. Further requirements for this research are specified in the following.

- The new approach should reuse SIP functionalities to provide the SOA "find-bind-execute" paradigm, because SIP has been standardised as the default protocol within the service stratum (call control) of IP-based telecommunication networks (e.g. ETSI NGN and IMS). The separation regarding to the upper layer, the application stratum, is not standardised and required, so there is no need to support further or other protocols than SIP. Normally an abstraction between the applications within the application stratum and the service stratum is not required, because the functionalities regarding to the applications such as registration, authentication and authorisation are already implemented within the service stratum (see section 2.1) and this is making use of SIP.

- SIP signalling should be used to implement a simple method for service provisioning as well as service composition.
- The framework should make use of an overlay for keeping data of registered subscribers and peers (e.g. application server). This overlay has to be based on a peer-to-peer technology. Further the services and their service descriptions should be stored within the overlay.
- Only minimal changes should to be implemented on SIP elements (e.g. User Agents, SIP Proxy).
- A mechanism to query services and service compositions is required. Therefore a simple syntax has to be defined for describing service compositions, so that end-users are enabled to search and use services and service compositions. Furthermore the end-users should be empowered to request service compositions and thus to (dynamically) create service compositions.
- Compatibility with existing SIP implementations of traditional legacy services (e.g. supplementary services) should be guaranteed.

3.5 Conclusion

This section introduced the foundations of Service-Oriented Architecture and its influence in telecommunications sector. The definitions and classifications of the term service have been discussed for SOA and telecommunications and a number of existing concepts for service composition have been compared, which vary in architecture, protocols and functionalities.

Also this section has illustrated that SOA concepts have a great impact within the telecommunication sector not only for the scope of management, because telecommunication systems are also service-oriented. The aim within the telecommunication sector today is to enable telecom operators and service providers to re-use existing services and capabilities in order to be able to provide value-added services.

It has been outlined that the demand for a service composition concept based on SOA and NGN aspects are constantly increasing. The different existing concepts of composing value-added telecommunication services have been compared with each other. The result has shown that the concept of NGSON fulfils the broadest spectrum of criteria, but some are missing, such as simplified architecture and interworking as well as minimising the signalling for service composition.

The main outcome of this chapter results in a set of requirements for the proposed approach for peer-to-peer based service composition which provides the basis for the framework introduced in chapter 4. The proposed approach has to fulfil the following.

- Service composition has to be user oriented. – The user has the ability to search and request service compositions once they are needed.
- Separation of service control and media transport. – This is realised through SIP, because it is only used for signalling, which means for the establishment of a session, e.g. a VoIP session. The payload has to be transported by other protocols (e.g. RTP).

- Openness for new services. – This implies that the framework has to be opened, so that new services and service compositions can be developed, offered, requested and utilised by anyone.
- Simplified architecture and interworking of the required features. – This facilitates the service composition, in contrast to the architecture defined for NGSON.
- Reduction of signalling. – To reduce the overhead of signalling for service composition.

4 Proposed service composition framework

Following from the requirements identified in section 3.4.5, this chapter introduces a novel service composition framework based on SOA concepts (see section 3.1) and SIP peer-to-peer infrastructure (see section 2.3) is suggested. The chapter starts by describing the SIP peer-to-peer architecture and its features (section 2.2), followed by the introduction of two new SIP elements (section 4.3 and section 4.4) which provide functionalities to support the integration of the so-called "find, bind, execute" paradigm (section 4.5). Furthermore the process of service provisioning (section 4.5.1), service discovery (section 4.5.2) and combining services (section 4.5.3) is illustrated.

4.1 SIP peer-to-peer architecture and its functionality

Based on the SIP specification (IETF RFC 3261, 2002) a hybrid P2P model can be realised. Basic SIP essentially supports P2P communication, because the specification already specifies combined client and server roles for the user terminal. According to (IETF RFC 3261, 2002) a hybrid SIP P2P model has already been defined (see section 2.2.1). Based on the typical SIP elements (User Agent, Proxy server, Registrar server and Location server) a P2P infrastructure can be realised. This infrastructure already enables direct peer-to-peer communication.

The SIP Registrar and SIP Proxy servers are only needed to lookup the location and actual addresses of the callee (SIP User Agent B). Therefore the temporary SIP URI is resolved to route the initial SIP requests and its response (INVITE and 200 OK). It

is obvious that the three-way-handshake (by sending ACK) is completed directly between the single peers respectively clients. From this point all data (payload and signalling) will be exchanged in peer-to-peer manner. Such a hybrid P2P infrastructure can be used to provide quick and easy services, especially value-added services; also end-users can offer these services. A SIP P2P overlay composed of SIP Registrar/Proxy servers has to be created as shown in section 2.3.3 (see Figure 2.6). The P2P overlay plays the role of the Location server (see Figure 4.1), as described in (Singh, 2006), this means that all the required data for location lookup and registration will be stored within the P2P overlay. The P2P overlay is based on a structured pure P2P model (see section 2.3.1), because it provides an efficient algorithm for proper searches, which are essential for location lookup in SIP. As already stated in section 2.3.1, structured peer-to-peer networks are mostly organised by distributed hash tables. Distributed hash tables are organised by key value pairs. Each key is unique such as a permanent SIP URI. By requesting for a key (e.g. permanent SIP URI) a value (e.g. temporary SIP URI) will be returned.

Based on this infrastructure, further SIP network elements can simply register with the SIP P2P overlay, for instance SIP application server, which is a novelty regarding to SIP-based networks. Normally servers (e.g. application servers) will not register themselves (Singh and Schulzrinne, 2004). The Proxy servers will be configured so that the application servers and their services are reachable. This implies that also this SIP application server needs to register with the SIP P2P overlay and that the application server is identifiable by a well-known (permanent) SIP URI (e.g. AS_One@bt.co.uk). This novelty allows registering any SIP network element.

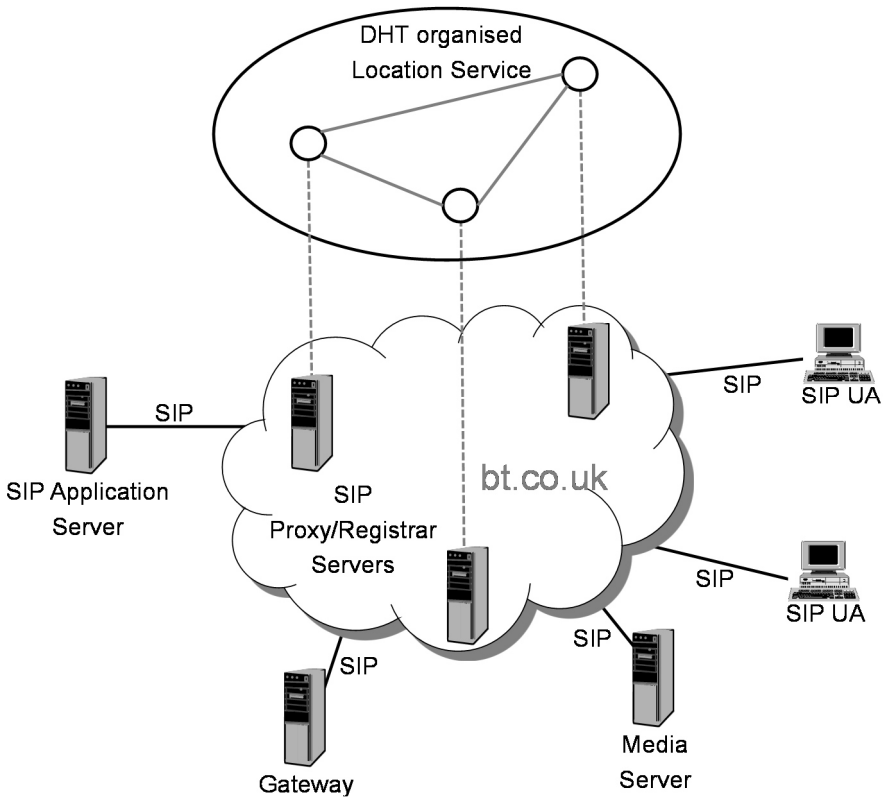


Figure 4.1: DHT based location Service

This architecture already offers the following benefits:

- Usage of specifications such as SIP (IETF RFC 3261, 2002), DNS (IETF RFC 1034, 1987) and (IETF RFC 1035, 1987), or dynamic DNS (IETF RFC 2136, 1997).
- Non-modified standard SIP User Agents can be used (based on the SIP specification).
- Fast lookups (resolving the temporary SIP URI from the permanent SIP URI) are better provided than by unstructured P2P models (Bischof, 2005) because of the retrieval strategy.
- Bottlenecks and single points of failure can be eliminated (due to P2P concept).

- Possibility to register and deregister any SIP network element, which is a novelty.
- Reduction of expenses (due to P2P concept), e.g. no need to continuously relay signalling or media through centralised servers.
- Scalability (due to P2P concept).
- Even end-users can offer services.

Essential features for service composition are not yet included in this architecture. To support the required features of service provisioning, service discovery and service composition, the architecture has to be supplemented with further functionalities. The following sections will illustrate the extensions.

4.2 Architecture enhancements for service composition

The described architecture from section 4.1 represents the basis of the proposed framework, the so-called service stratum building the fundament for the application stratum (see Figure 4.2). Within the application stratum, the missing functional elements for service discovery and composition are implemented. These elements are the service discovery server and service composition server, which will be described more deeply in the following sections 4.3 and 4.4. The discovery server is responsible for service discovery and publishing, it supports mechanisms to find services by given criteria, such as keywords describing the service functionality, and store service publications within a repository, such as storing service descriptions in a database.

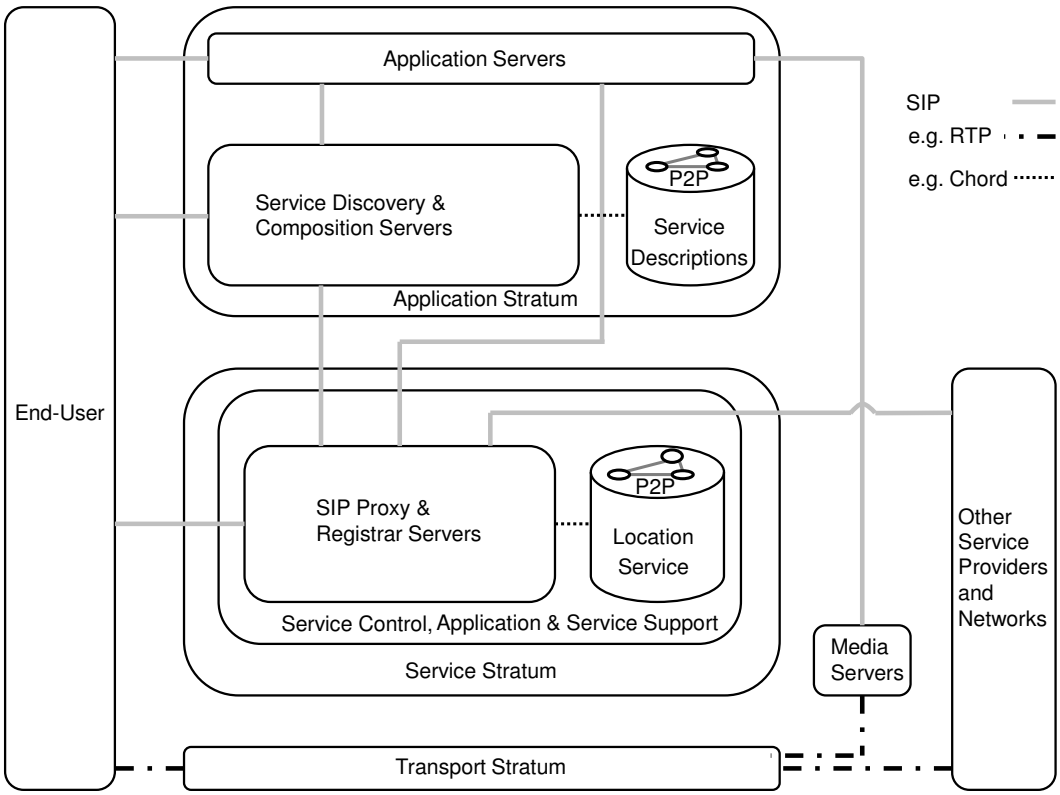


Figure 4.2: General framework architecture

As shown in Figure 4.2, the service discovery and service composition servers are connected to a repository that holds the service descriptions. This repository is built on a DHT where the service descriptions are stored and may use Chord (Stoica *et al.*, 2001) as an underlying protocol.

In general the framework's architecture is leant on the architecture of NGN (compare Figure 4.2). The framework's architecture fulfils the characteristics of the NGN concept, which are related to services. These characteristics are that the framework is packet-based, because it uses protocols such as SIP or RTP, which are IP-based. It follows the requirement for separation of call/service control from the media data transport, supports application servers and multimedia services. Moreover, the architecture of the framework provides unrestricted access for users to different

networks and service providers and finally, the whole framework is scalable, because of the separation of the different functions from the strata. If there is a need for more application servers, they can simply added without any changes in the other strata this extends to the SIP Proxy and Registrar servers.

Through SIP interfaces the SIP Proxy and Registrar servers are connected to the end-users for registration, authentication and usage of the teleservices (e.g. telephony and instant messaging) and supplementary services (e.g. call transfer) provided by the service stratum, the application servers for service invocation, registration, authentication and registration of services (value-added services), the discovery and composition servers regarding service and composition queries and possibly other service providers. The service discovery and composition servers are also accessible by SIP interfaces, which are used by the application servers to publish their service descriptions. The end-users may directly connect the service discovery and composition servers for searching services and requesting compositions. The services and the composite services are invoked directly by the end-users through the SIP interface at the application servers. The application servers may control media servers to enable multimedia services, for instance audio/video conferences. The presented SIP peer-to-peer architecture from section 4.1 has been enhanced with the functionalities for service discovery and composition, because it is not supporting the named functionalities by its own. The described application stratum can be seen as an implementation of a service delivery platform as described in section 3.1.2. The so-called service exposure layer of the service delivery platform within this architecture could be implemented on the application servers. The SOA principles, loosely coupling, location independency, reusability, discoverability, composability of

services and the possibility to publish service interfaces, of the service orchestration and management layer are implemented by the service discovery, composition and application servers. Furthermore the usage of telecommunication resources is enabled by SIP as an interface. One difference has to be distinguished. No abstraction from the lower telecommunication resources, e.g. location service, is needed, because within the service delivery platform, SIP is used for service composition and control.

In relation to other service delivery platforms, the novelty is the utilisation of SIP within the service delivery platform, which supersede the abstraction from the underlying telecommunication infrastructure. So no further protocols are necessary within the service delivery platform. Other service delivery platforms are built upon middleware technologies like JAIN SLEE or based on Java EE (Enterprise Edition). This mostly implies that the service implementations are located on the same machine or that other technologies are used for communication between the service implementations, e.g. JMS (Java Message Service) or web services.

Another difference between existing implementations and concepts (see section 3.4) is the way services are invoked. Based on the presented framework the services are invoked directly by the end-users in contrast to others, where a central node (e.g. a SIP Proxy server) is responsible for that. Also the NGSON are making use of a centralised element, the so-called Service Routing Functional Entity (see section 3.4.3).

By using distributed data bases for the user's location and service descriptions within the framework's architecture, single points of failures are eliminated. This means that the DHT-based solutions of location service as well as the service description repository have implemented algorithms to provide redundancy. In other words, if a

peer of the implemented DHT breaks down or goes offline, the stored data of this node is shared already with other peers of the DHT for redundancy aspects. After detecting that the peer broke down, the redundant data is newly distributed, so that the data is persistent even if other peers will leave the DHT.

Two further features substantiate the efficiency of the proposed framework. Regarding to other solutions, for instance NGSON, the complexity of the overall solution has been reduced, that is the reduction of necessary functional entities as well as the removal of abstractions, e.g. from the underlying networks. Furthermore the necessary amount of signalling has been reduced, this will be explained further in section 7.4. For example when utilising the concept of NGSON or Service Broker the amount of messages, if the entities will communicate with a signalling protocol, is higher than the amount of messages within the proposed framework, because of the utilisation of a centralised network element, e.g. the Service Routing Functional Entity or Service Broker.

4.3 Service discovery server

This section describes the novel service discovery function that is implemented as part of a service discovery server, which represents the service directory. Within the service directory different information is stored similar to, for example, the UDDI service, which is known from the web service technology. The stored information is enclosed within a service description language based on XML. The service description contains data such as the service name (e.g. a keyword), a human readable service description, a service identifier (e.g. a service URI), other metadata representing the categorisation

of the service and a service interface whereby the service can be used. Chapter 5 will specifically discuss the design and implementation of the service description.

The logical function of the service discovery server can be integrated with the SIP Proxy/Registrar server on one physical device. This implies that also the service discovery server is directly reachable by the domain just as the SIP Proxy/Registrar servers (see Figure 4.1) and their functionalities. As a central functionality the service discovery server supports basic features from the SOA providing the publish and find features of the "find, bind and execute" paradigm as illustrated in Figure 4.3, so the service discovery server is the core element of the service triangle. SIP application servers can publish their hosted services at the discovery server which can then be requested for invocation by SIP User Agents.

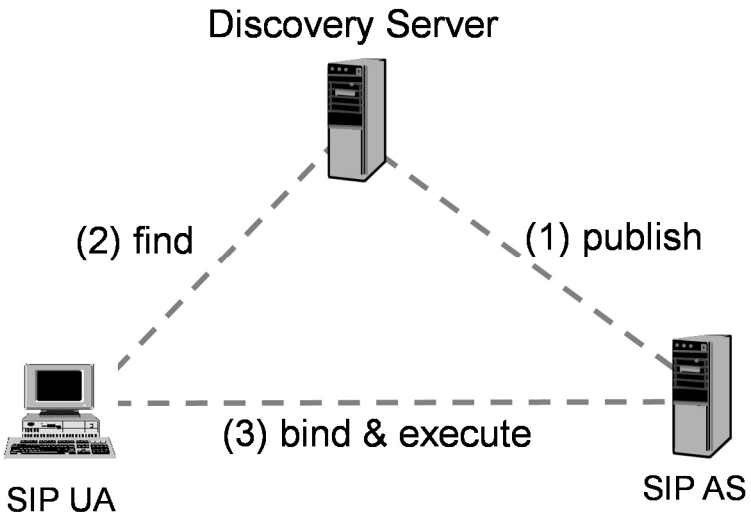


Figure 4.3: Service triangle of the framework

The published service descriptions need to be stored within the DHT (location service). Therefore a database structure must be implemented for storing different competing service descriptions under one keyword, e.g. conference or answering machine, which

means that behind the keyword conference several services can be found and therefore also the corresponding service descriptions. This implementation guarantees that service descriptions can be requested by a keyword. The following feature list summarises the properties of the service discovery server.

- Plays the role of the service repository.
- Facilitates storage of service description publication.
- Enables the query for a service based on keywords.
- Makes use of DHT as database, for storing service descriptions.
- Allows for detection of services and service descriptions by keywords.
- Queries are extensible, e.g. other criteria may be used such as service costs or quality.
- Replies service requests by the return of a list, which contains all possible service identifiers (here SIP URIs).
- Represents a central feature to provide service publication and query.

The innovations presented within this section are the linking of a service discovery function, here represented by the discovery server, and a DHT-based database storing the service descriptions and the possibility to search for services by keywords.

4.4 Service composition server

This section introduces the features of the service composition server, which generally creates composed services out of published services by their service descriptions. The composition server is the missing link to achieve service compositions according to the SOA feature “Composable”.

The logical function of the service composition server can be integrated within the same physical entity along with the discovery server and SIP Proxy/Registrar server. In Figure 4.4, the functional architecture of the framework is illustrated showing the different server functions (SIP Proxy, SIP Registrar, Discovery and Composition), communication interfaces (SIP and P2P) and the overlay with its features (location service and database storing the service descriptions).

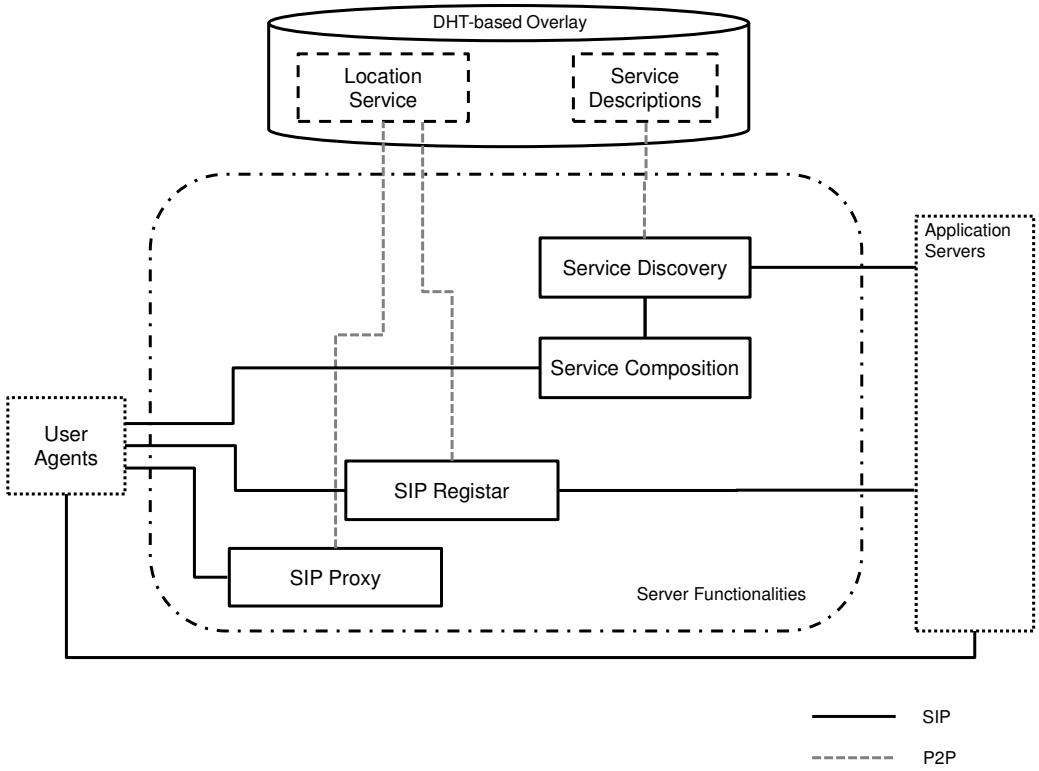
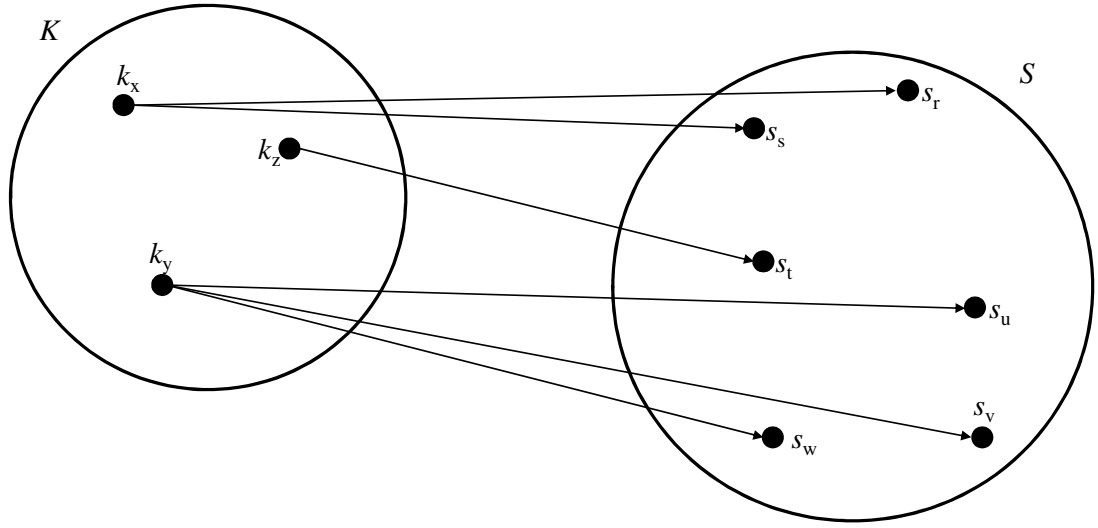


Figure 4.4: Functional framework architecture

To provide service composition, the composition server cooperates closely with the service discovery server in order to create compositions on the basis of service descriptions. For that purpose, the composition server requests keywords representing services at the discovery server. The keywords will be extracted from the request a subscriber has sent to the composition server. The request contains a sequence of

keywords describing a service composition. Assuming that for each keyword there is at minimum one existing service, this means that a relation *R* between an element *k* (keyword) of the set *K* (set of all keywords) and element(s) *s* (service) of the set *S* (set of all services) can be identified. An exemplary relation is illustrated in Figure 4.5. The request for a service composition is formed by a sequence of keywords which are linked by the symbol "|", e.g. $k_x \mid k_y$. This symbol specifies the catenation of the services hence a directed graph is the result, e.g. $k_x \rightarrow k_y$. This formal description is derived from the Unix pipes used for pipeline of commands (Ritchie, 1984).



$$R = \{(k_x,s_r), (k_x,s_s), (k_y,s_u), (k_y,s_v), (k_y,s_w), (k_z,s_t)\}$$

Figure 4.5: Exemplary relation of keywords and services

Due to the metadata extracted from the service descriptions that are returned by the discovery server as a result of the keyword query, the service composition server is able to build composite services. For this the composition server extracts necessary information out of the metadata in order to get to know if the services are connectable. This depends on the in- and outputs of the services. The data type of the services' in- and outputs must be equal, for instance an output of the data type audio/speech is only

connectable with an input of the same data type. The result of the composition process is a set of directed graph trees, where each tree is a service composition. The composition server will reply to the subscriber's request right after service composition creation. This reply contains a list of compositions that consist of chains of SIP URIs. The SIP URI chains will be explained in section 4.5.3 in more detail. With the received list of compositions, including different SIP URI chains, the subscriber has all the information to choose and invoke a preferred composition.

However a prioritisation of the available service compositions helps the user to make an optimal choice. If the number of possible compositions is high, then the list may be limited. Therefore the subscriber should be able to set a keyword for the prioritisation. This keyword is a non-functional property regarding to services and service compositions. Exemplary keywords are quality, costs and bandwidth. The following list of features will summarise the functionalities of the service composition server.

- Composes services out of published and registered services.
- Enables the query for service compositions.
- Cooperates closely with the service discovery server.
- Extracts keywords out of a subscriber's composition request.
- Sends requests with keywords to the discovery server that will be answered by returning a set of service descriptions for the keywords.
- Analyses the service descriptions to identify the in- and outputs and their data types of the services.
- Creates SIP URI chains representing service compositions by concatenating service in- and outputs.

- Sorts the resulting service compositions by non-functional properties.
- Replies to the subscriber's request with a sorted set of SIP URI chains.

4.5 Implementing the "find-bind-execute" paradigm in a SIP-based environment

This section depicts the implementation of the "find-bind-execute" paradigm within the proposed SIP-based peer-to-peer infrastructure. As already mentioned in section 4.1, the proposed infrastructure has to be extended with further functionality to provide service discovery, provisioning and composition (see Figure 4.6).

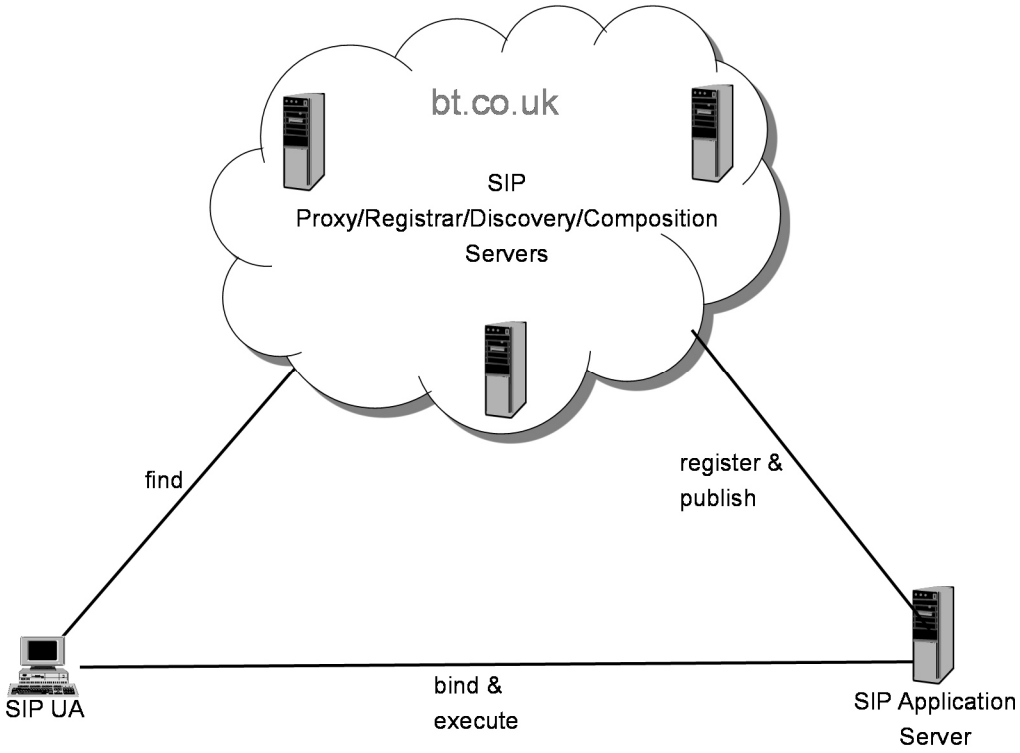


Figure 4.6: SIP-based find-bind-execute paradigm

The presented methods register, publish, find and bind are based on SIP. The register and publish methods are different. Within SIP two messages exists: one is the REGISTER method, which is used to register and deregister SIP network elements as

well as services, the other one is the PUBLISH method which is used to publish the service descriptions. The following sections will describe the SIP-based communication.

4.5.1 Service provisioning in SIP-based peer-to-peer network

To offer access for end users to the new services implemented in SIP application servers, application servers must be extended by a further functionality. An application server uses the request REGISTER, such as a SIP Digest (IETF RFC 3261, 2002), in order to register itself and its services with the network. To sign on implemented services the same request as shown in section 2.2.1 is used (Schmidt *et al.*, 2006), (Lehmann *et al.*, 2008a) and (Lehmann *et al.*, 2008b). The URI to use for registration and authentication of a service respectively originates from the subdomain model (e.g. "sub.domain.co.uk"). The following example will clarify the usage. Assuming that an application server has registered the following URI "as1@domain.co.uk", he is authorised to send the following SIP request to the SIP Registrar, whose domain is resolved by DNS, to register serviceX:

```
REGISTER sip: domain.co.uk SIP/2.0
From: <sip:serviceX.as1@domain.co.uk>;tag=1234
To: <sip:serviceX.as1@domain.co.uk>
Contact: <sip:serviceX.as1@88.88.88.88>
Expires: 3600
```

After registering the new service, it is reachable by its URI and subscribers can make use of it. By using the model presented in Figure 4.7, registered application servers can provide any number of services. The SIP header field "Expires:" sets the period of time for the validity of the registration that in this case is valid for one hour. As a result the application server has to register the service in defined intervals of time to set the

service accessible. With the same SIP request the service can also be deregistered from the network, setting the SIP header field "Expires:" to "0". The presented registration process is without authentication.

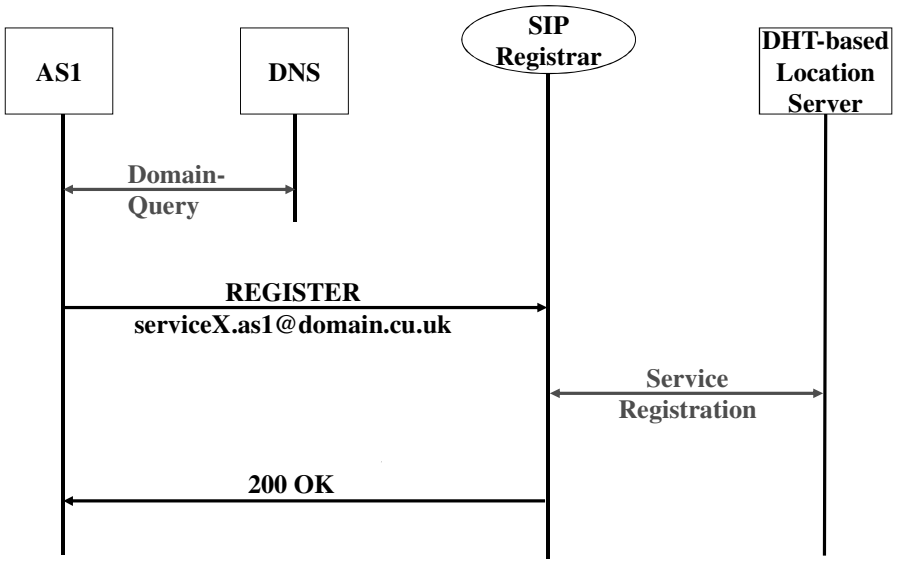


Figure 4.7: SIP-based service registration

SIP User Agents can then access the provided services hosted by the application servers or other application servers can make use of them as this mechanism fully supports "openness for new services". Also some security features are supported together with SIP (e.g., authentication, SIPS (Session Initiation Protocol Security) using TLS (Transport Layer Security)).

4.5.2 Service discovery, binding and publishing in SIP-based peer-to-peer network

The service discovery server has to be extended with new features, which are explained in section 4.3, so that the framework's architecture supports publishing and discovery of services. During the past years, the IETF has standardised SIP and several extensions, particularly the SIP event framework (IETF RFC 3265, 2002) as an

architecture for status delivery from and to any host in IP-networks and the SIP extension for event state publication (IETF RFC 3903, 2004). Using these standards it is possible to create such a service provisioning architecture.

As part of the proposed architecture, new services and the associated service descriptions will be published using the SIP PUBLISH method. The body of this SIP method contains the service description (such as the service name, human readable service description, service id, and service URI) specified using XML (Rahman and Buford, 2006). The PUBLISH method is sent to the overlay network, where the SIP Proxy and Registrar servers are extended by the SIP discovery server regarding to, called SIP event server (Trossen and Pavel, 2005), to support the publishing and discovery of services. All information belonging to the events is hosted by the SIP event server, called SIP discovery server from now on. Figure 4.8 demonstrates the SIP implemented “find-bind-execute” paradigm.

First a SIP AS publishes a service description with the SIP method PUBLISH, then this service can be requested with as SIP SUBSCRIBE method. The service request will then be answered with a SIP NOTIFY method containing the necessary information to bind and execute the requested service with a peer-to-peer invocation, realised by utilisation of the SIP three-way handshake (INVITE/200/ACK).

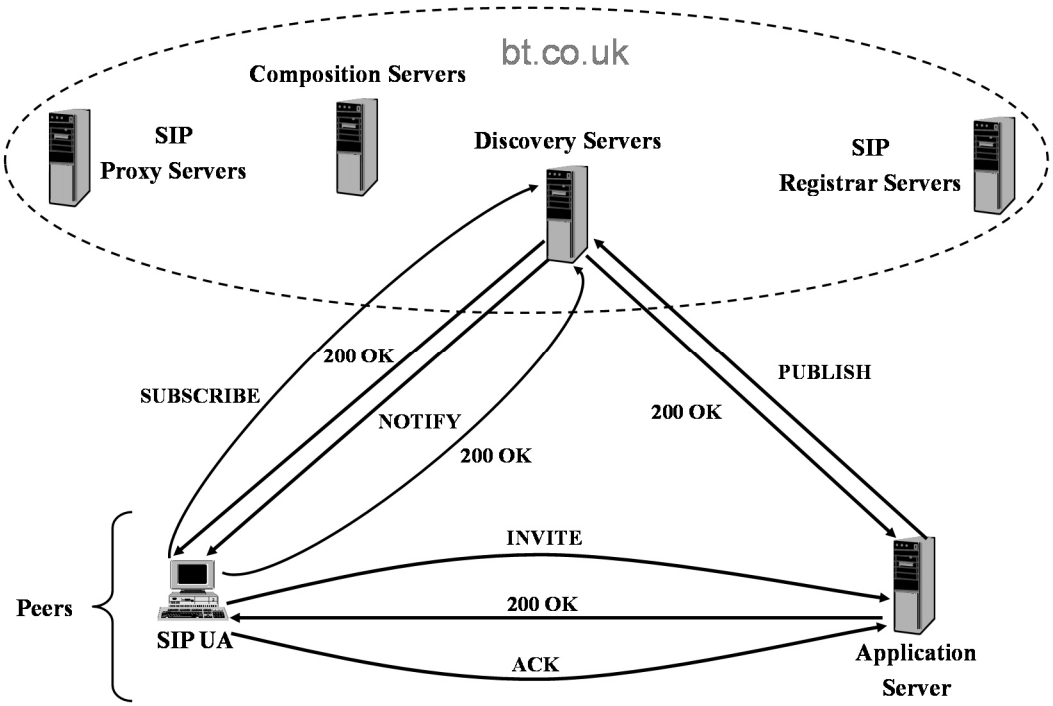


Figure 4.8: SIP service discovery and publishing

The message body of the SIP Publish method contains the service description. An exemplary service description is presented as follows.

```
PUBLISH sip:serviceX.as1@ bt.co.uk SIP/2.0
To: < sip:serviceX.as1@ bt.co.uk >
From: < sip:serviceX.as1@ bt.co.uk >;tag=1234
Expires: 6000
Event: x-servicedesc
Content-Type: application/servisdesc+xml
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>
<Service xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

xsi:noNamespaceSchemaLocation="../xsd/ServiceDescription.
xsd">
  <ServiceId id=serviceX.as1@bt.co.uk/>
  <ServiceGoal>
    <Keywords>
      <Keyword>conference</Keyword>
    </Keywords>
  </ServiceGoal>  …
</Service>
```

To implement the discovery functionality two further SIP messages are used, namely SUBSCRIBE and NOTIFY. The SIP message SUBSCRIBE will be used to do a service query and the SIP message NOTIFY will be used to answer the service query. In the following the service discovery is described. A subscriber (here: SIP UA) sends the SUBSCRIBE message to initially subscribe to an event as service query and receives NOTIFY for the initial notification, which contains the service information, and all subsequent NOTIFY messages that relate to this subscription. Each subscription carries a desired lifetime of the subscription. In case of a lifetime of zero, the SIP discovery server merely sends back all available and matching services at this point of time. If the lifetime is greater than zero, an availability subscription to services in the future is established (Rahman and Buford, 2006) and (Trossen and Pavel, 2005). As an example, the SIP messages SUBSCRIBE and NOTIFY are shown below.

```
SUBSCRIBE sip:servicerequest@ bt.co.uk SIP/2.0
To: < sip: servicerequest @ bt.co.uk >
From: <sip:userA@ bt.co.uk >;tag=4567
Event: x-servicedesc
Accept: text/plain
Content-Type: text/plain
Expires: 0
Content-Length: ...

conference|newsticker

NOTIFY sip:userA@ bt.co.uk SIP/2.0
To: <sip:userA@ bt.co.uk >;tag=4567
From: < sip: servicerequest@ bt.co.uk >
Event: x-servicedesc
Content-Type: application/ xml
Content-Length: ...

<sip: audioconference.telekom@ bt.co.uk>,
<sip: newsticker.bbc@ bt.co.uk>
```

The illustrated SIP NOTIFY contains a SIP URI chain that represents the sequence of service invocation of a service composition. The answer must not necessarily contain a SIP URI chain, also a single SIP URI could be returned. Beside that information every service or service composition is described briefly by a human readable description that is also part of the SIP NOTIFY's body.

The resulting architecture (see Figure 4.8) matches to the find, bind and execute paradigm of the SOA. The already illustrated service triangle presented in section 4.3 can be integrated into a general NGN strata view (see Figure 4.9) (Lehmann *et al.,* 2009a). The SIP discovery servers are representing the service directory. The service provider is the AS and the service customer is the SIP User Agent (SIP UA).

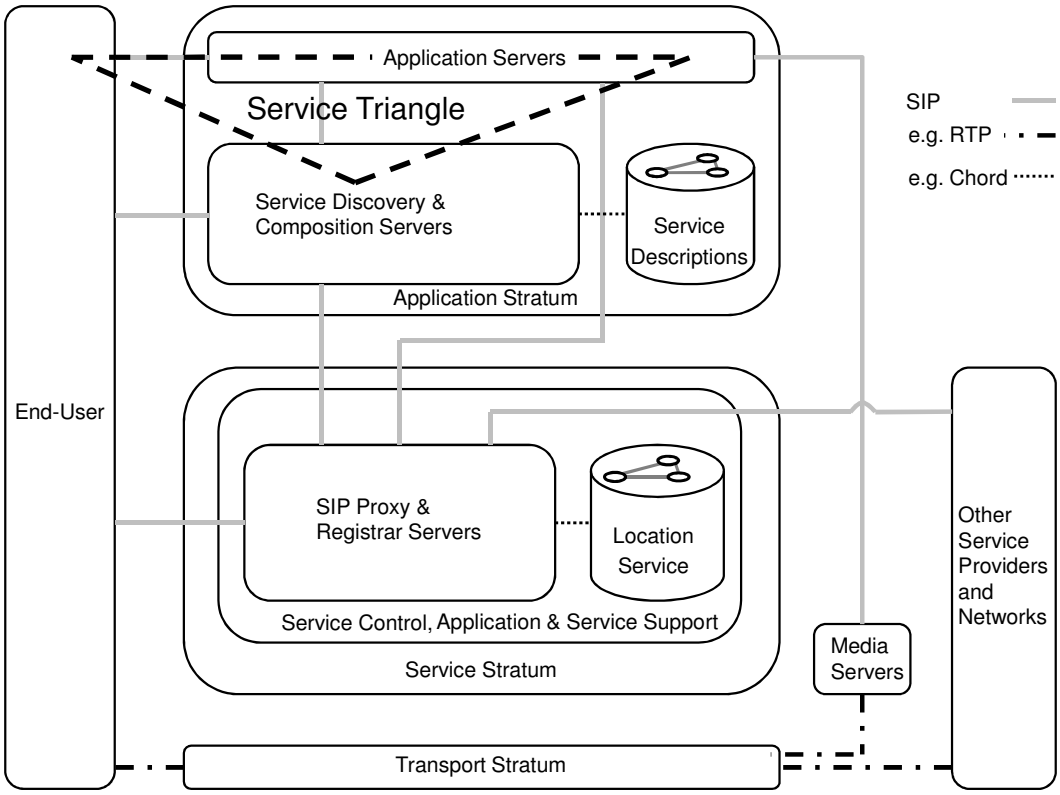


Figure 4.9: Service triangle in general NGN strata view

The described mechanism of service publishing and discovery, derived from (Rahman and Buford, 2006) and (Trossen and Pavel, 2005), will fulfil the needs for service discovery given from the SOA concept. The novelty of the presented mechanism is that it is based on P2P and the relation to SOA has been referenced.

4.5.3 Service composition in SIP-based peer-to-peer network

Based on the explained approach within this chapter, the combination of distributed services can be implemented easily. Therefore the SIP Route header (IETF RFC 3261, 2002) and (MSF IA SIP 006, 2005) will be used for invoking the service. This SIP header field holds a set of SIP URIs, the previously mentioned SIP URI chain, which represents the addresses of the single services. The concept proposed by the MultiService Forum (MSF) is based on a centralised architecture referred to IMS, as already stated in section 3.4.1. The service identification and routing will be organised by a Service Broker. In the following the combination of services will be explained by a simple example.

As a sample, a service combination can be an audio/video conference with an embedded news ticker (e.g., actual football scores). Granted that all these services are single distributed services (reusable services) every single service is addressable by a permanent SIP URI (e.g., audioconference.telekom@bt.co.uk). The composed service can be achieved by chaining the SIP URIs. The resulting service could be described by the following SIP message (note only the relevant header fields are shown).

```
INVITE sip: videoconference.bt@bt.co.uk SIP/2.0
...
Route: <sip: audioconference.telekom@bt.co.uk;lr>,
            <sip: newsticker.bbc@bt.co.uk;lr>
...
```

It is obvious that the services have to be previously registered and published at the framework as already described within this chapter, so that these services can be used. The SIP message would be routed due to the Route headers as shown in Figure 4.10.

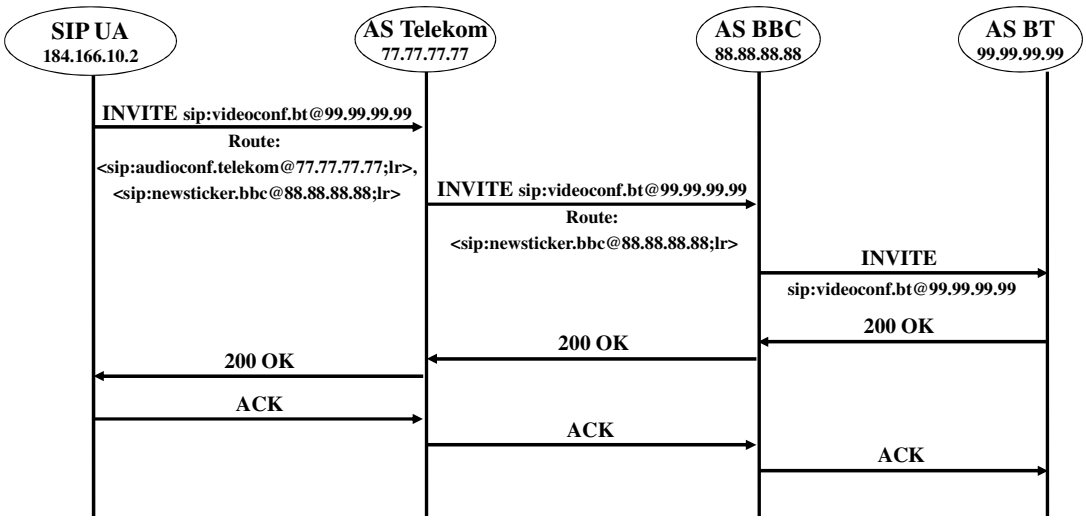


Figure 4.10: Service composition example with SIP Route header

As depicted in Figure 4.10 the SIP message INVITE is forwarded from one AS to another. This requires that during the composition process the permanent SIP URIs are resolved into their temporary SIP URIs. This could be achieved by the SIP Proxy server in connection with the composition server that will create the service compositions. While routing the SIP message INVITE from one AS to another, the invoked services themselves may convert parts of the SIP message, and the news ticker service might modify the SIP body of the initial message. For the purpose of embedding the news ticker into the video stream of the videoconference, the SDP media description part, containing video information (see Figure 4.12), has to be modified by the news ticker service, so that all video data will be forwarded to the news ticker. As a result, the news ticker service can overlay the ticker information into

the video data. The resulting combination of the news ticker and videoconference could look like presented in Figure 4.11.

Figure 4.11: News ticker embedded in video conference

This can be achieved by modifying the corresponding information (here: `m=video`) from the SIP body as follows.

```
…
v=0
o=client 1234 0 IN IP4
184.166.10.2
s=ServiceCombination
t= 0 0
c=IN IP4 184.166.10.2
m=audio 1000 RTP/AVP 0 8 3
m=video 2000 RTP/AVP 34
…
```

Figure 4.12: SDP media description part

The submitted IP address given by the connection data parameter *c* and the submitted port given by the media description parameter *m* (which in Figure 4.12 has the value 2000, indicating that the video will run on port 2000) specify the socket the subscriber will expect the video data. The news ticker service adds a new *c* parameter to the SDP

body beneath the *m* parameter (video) with its own IP address, before forwarding the SIP request, so that he can receive the video data. Figure 4.13 will illustrate the modification within the SDP part of the SIP request before forwarding to the videoconference service. Also the RTP media transport is depicted in Figure 4.13.

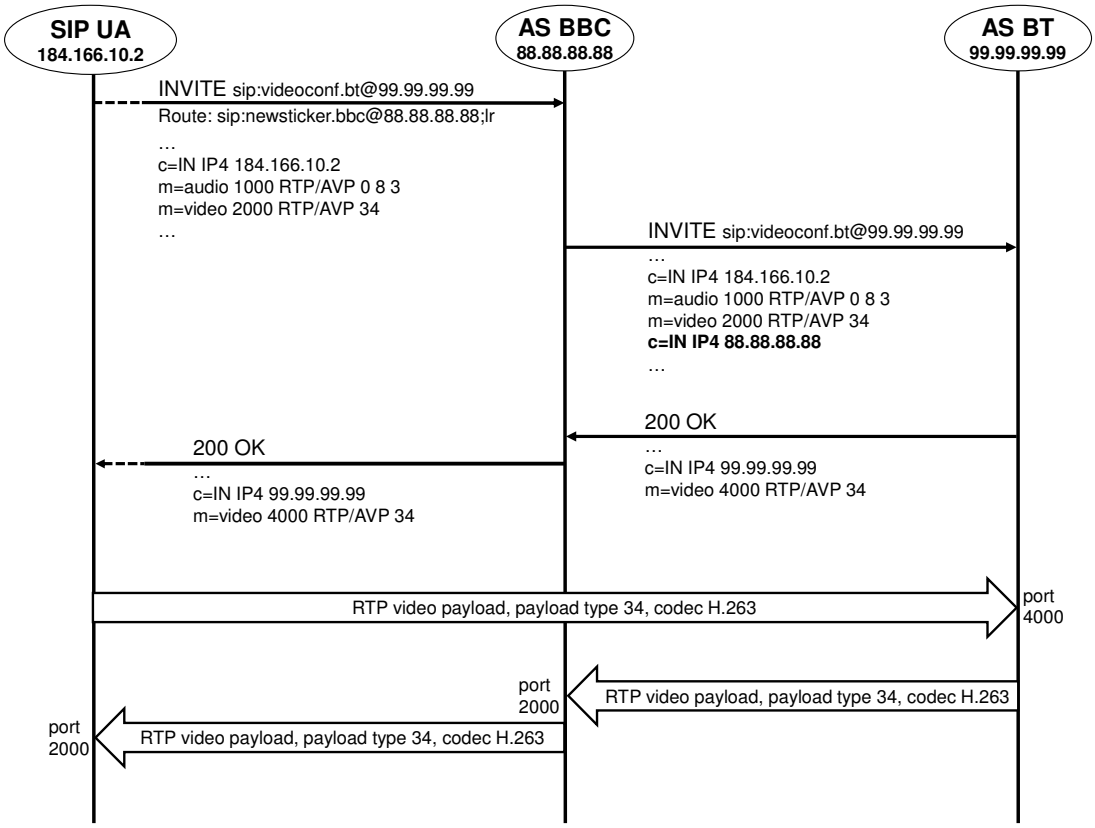


Figure 4.13: Modified SDP media description part

The video data is directly sent to the videoconference service by the SIP User Agent to port 4000, on which the service is listening for incoming media. The videoconference media data is sent by the service to the news ticker service as offered within the SDP part, which was forwarded by the news ticker service to the videoconference. The news ticker service now decodes the video data and embeds its news ticker as shown in Figure 4.11. After embedding the news ticker data the news

ticker service encodes the video and sends it to the SIP User Agent to its offered socket from the initial SIP request (here: 184.166.10.2:2000).

By modifying the SDP message structure as described, the video payload can be routed as intended and the combining of services can be easily implemented. This brief example shows the possibility to combine services easily, including the novel aspect that services may modify the media descriptions during session initiation to route the media data. Media data routing as well as termination, conversion and transcoding is possible.

Summarising, the SIP-based procedure can be described as a service combination by chaining services with SIP, using the SIP Route header. In (IETF RFC 3261, 2002) the Route header field is used to force routing for a request (e.g. INVITE) through the listed set of SIP URIs. The request is sent to the location derived from the topmost value of the Route header. The SIP entity that receives the request, removes the topmost value, if it is representing its own URI. If a previously second SIP URI, now topmost, is present in Route header, the request is forwarded to that address; otherwise the Request URI is used. Every following SIP entity receiving the request has to handle the same way (see Figure 4.14).

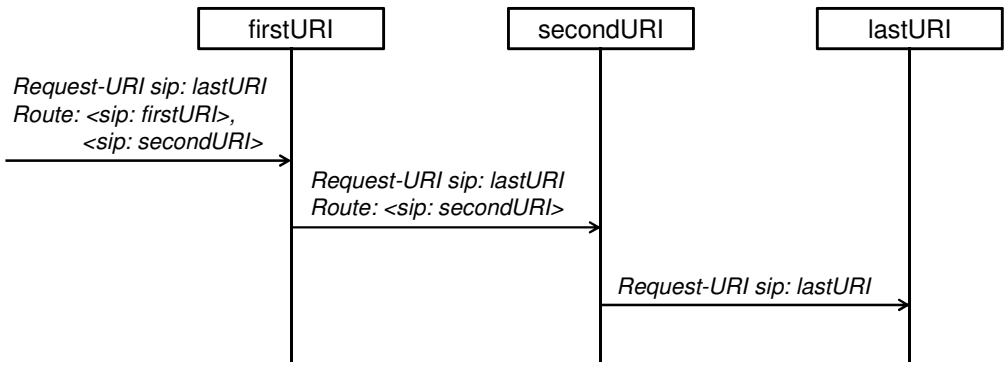


Figure 4.14: General SIP request routing with Route header

The services themselves can modify SIP headers and the body (e.g. SDP) of the request depending on their implementation, as already illustrated by the example in this section.

4.6 Conclusion

This chapter introduced a SIP-based P2P framework for service composition that fulfils the requirements of service location independency, service discoverability and service composability according to SOA, as outlined in chapter 3. The location independency and further requirements such as accessibility, availability and service abstraction are covered by the SIP-based P2P model itself, as outlined in section 4.5.1. Service discoverability is covered by the SIP discovery server and the described service discovery and publishing based on SIP, as illustrated in sections 4.3 and 4.5.2. The service composability is achieved by the service composition server and the specified SIP signalling, all described in sections 4.4 and 4.5.3.

Two novel functional entities have been introduced, namely SIP discovery server and service composition server, also the utilisation of SIP to support service provisioning, discoverability, publishing and composition has been specified. Following this approach, the advantages given by the framework are inherited from the P2P architecture (e.g., scalability, cost reduction, fault tolerance) and the support of standardised SIP network elements (e.g., SIP User Agents and SIP Proxy/Registrar servers).

A foundation has been defined for a service composition framework, but a number of requirements remain to be specified, most notably the service description. Without a

machine readable service description an automated service composition is impossible, hence the following chapter will investigate the alternatives for describing service properties.

5 Service Description Language (SDL) and algorithm for service composition

A well-defined service description language is the basis for automated service composition because the network element, which will create service compositions, needs to have knowledge about the services and their interfaces so that it can combine them. The service description language has to be machine readable and interpretable. Section 5.1 will illustrate different service description languages for telecommunication services. In section 5.2 a novel service description language is presented. The different discussed service description languages will be compared against each other in section 5.3. To enable automated service composition based on a machine readable service description language an adequate algorithm is needed which will be presented in section 5.4.

5.1 SDLs in telecommunication

This section will discuss different service description languages (SDL), which are proposed for utilisation especially within the telecommunication sector:

- SPATEL (SPice Advanced service description language for TELecommunication services) (Cordier *et al.*, 2006)
- Composition description language for service composition in NGN environments (Bether, 2008)
- A new approach to classify and describe telecommunication services (Lehmann *et al.,* 2009b)

The result of the discussed SDLs is a novel SDL which is used for implementation within the proposed framework.

Beside these service description languages others exist, which will be discussed here briefly.

- JSDL (JSON Service Description Language): It is a JavaScript Object Notation (JSON) –based interface description language that is used to describe functionality offered by a HTTP web service. (JSDL, 2014)
- WADL (Web Application Description Language): The specification (WADL, 2009) describes the Web Application Description Language. An increasing number of Web-based enterprises are developing HTTP-based applications. Typically these applications are described using textual documentation that sometimes supplemented with more formal specifications such as XML schema for XML-based data formats. WADL is designed to provide a machine processable description of HTTP-based web applications.
- RSDL (RESTful Service Description Language): The RSDL is a machine- and human-readable XML-based description of HTTP-based web applications. (Robie *et al.*, 2013)
- WSDL (Web Services Description Language): WSDL describes web services by their network endpoints. WSDL is XML-based. It defines service network endpoints and message types in an abstract way. It is extensible and not bound to concrete network protocols. (Chinnici *et al.*, 2007)

All of the briefly mention service description languages, JSDL, WADL, RSDL and WSDL are not especially defined for telecommunication services. Furthermore the

first three ones are specified to describe only HTTP-based services. This research work is focused on SIP-based services, so only the service description languages focused on telecommunication will be discussed in the further sections.

5.1.1 SPATEL (Spice Advanced service description language for TELecommunication services)

SPATEL was developed within the IST-SPICE (Information Society Technologies-Service Platform for Innovative Communication Environment) project (Cordier *et al.*, 2006) and it allows the specification of services in a platform independent manner, including annotations concerning semantic and non-functional properties (Silva *et al.*, 2007). The SPICE services are composed based on a collection of components, whose services can be published and used in service compositions by end-users and application developers. This is made possible by applying web services technology and the Service-Oriented Architecture principles. Within the SPICE project, an Automatic Composition Engine (ACE) was developed, which supports end-users and application designers on the development of service compositions. The composition engine expects to receive either service requests from the end-user or from the developer and delivers a service composition matching the request or it will send a list of alternative compositions. The approach of the SPICE project relies on the use of semantic annotations on atomic services, which are services that cannot be decomposed into more granular services, and on service requests, to perform the service discovery, matching and composition.

To obtain service compositions automatically, the service request and description of services and composite services need to be annotated with some semantics, by using

ontologies. OWL-S (Ontology Web Language for Web Services) (Martin *et al*., 2004) and SPATEL allow the definition and creation of semantic annotated (web) services. The SPICE ACE is built-on four basic elements: Semantic Analyser, Composition Factory, Property Aggregator and Matcher (see Figure 5.1).

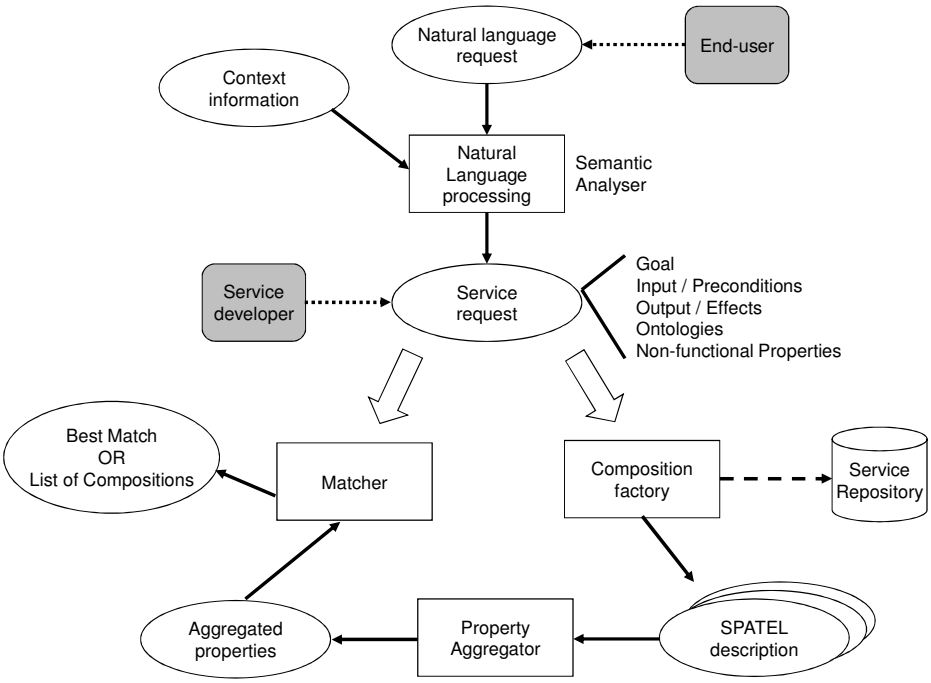


Figure 5.1: SPICE ACE architecture (Silva *et al.*, 2007)

The Semantic Analyser is used to transform end-user requests into a formal service request, which corresponds to requests given by a service developer. The services and service requests are characterised by functional and non-functional properties. Functional properties include the service goals (e.g. translate a text), inputs (e.g. text and language), outputs (e.g. translated text), preconditions (e.g. the text must be in a special format) and effects (e.g. text is translated). These properties are used to perform the service discovery, matching and composition. Non-functional properties used to limit the space of compositions that fulfil the service request, and to rank the generated

set of compositions and may include cost and security. The properties are defined by ontologies following the OWL-S specification (Martin *et al*., 2004).

The Composition Factory queries the service repository for a service that matches the service request; if such a service exists, the matching service is returned. In case no matching service is found, the Composition Factory creates a composite service that resolves the request and may generate a list of alternative compositions to match the request. The produced compositions are passed to the Property Aggregator, which computes the non-functional properties and aggregates them per atomic component service, such as that the QoS parameters are summed up for each composition. The generated composite services are passed to the Matcher, which allocates the composed services to the requested service using their non-functional properties.

Figure 5.2 illustrates the structure of the service description in a top level view. A service itself is described by three properties “presents”, “describedby” and “supports” where these properties are defined by the classes ServiceProfile, ServiceModel and ServiceGrounding (Mika *et al.*, 2004).

According to (Martin *et al*., 2004), the ServiceProfile provides the information needed to discover a service which organisation provides the service, which functionality the service computes, and a set of properties that are used to describe features of the service. The service providers’ information contains data concerning e.g. the service providers’ company name. The functional description specifies the input(s), output(s), precondition(s), and effect(s) (IOPEs).

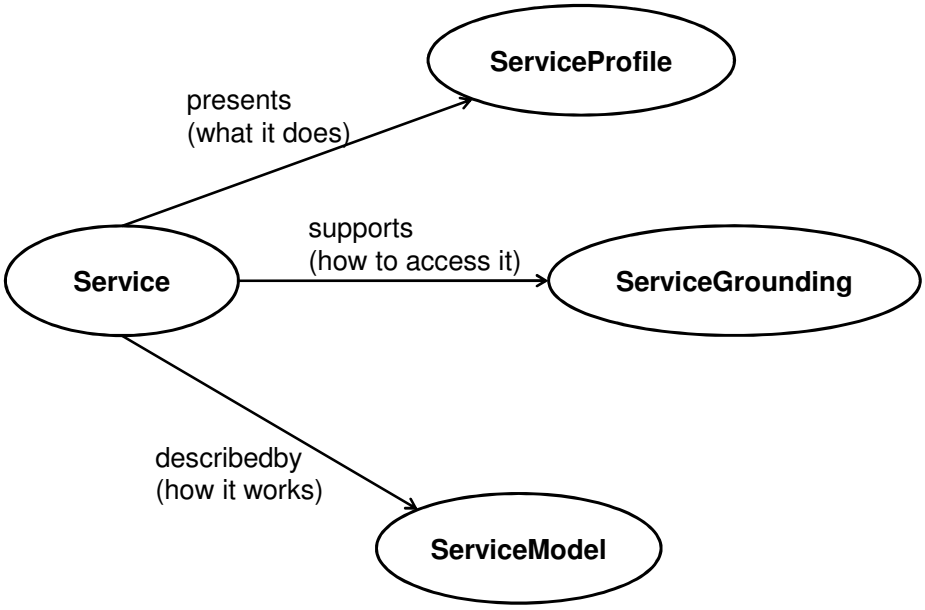


Figure 5.2: Top level of the service ontology (Martin *et al.*, 2004)

For example, a selling service may require as a precondition a valid credit card and as input the credit card number. As output it generates receipt, and as effect the card is charged. The last variable within the ServiceProfile contains information about quality and a categorisation of the service, for example, the category of the service within the United Nations Standard Products and Services Code (UNSPSC) classification system (UNSPSC, 2013). The UNSPSC is a hierarchical convention that is used to classify all products (e.g. telecommunications cable) and services (e.g. telecommunication service). Such a UNSPSC consists of five layers (see Table 5.1).

Table 5.1: UNSPSC hierarchy (according to UNSPSC, 2013)

Hierarchy	**Category Number**	**Name**
Segment	81	Engineering and Research and Technology Based Services
Family	16	Information Technology Service Delivery
Class	17	Telecommunication Service
Commodity	11	Videoconferencing Service
Business Function	XX	Function performed by an organisation in support of the commodity

Each layer is translated into a 2-digit code. For example a videoconferencing service is identified by the code 81161711. The first two digits describe the segment (here: 81-"Engineering and Research and Technology Based Services") the following two specify the family (here: 16-"Information Technology Service Delivery") followed by the class (here: 17-"Telecommunication Services") and the commodity (here: 11-"Videoconferencing Service"). Finally the last two digits are describing a special business function performed by an organisation in support of the commodity. They are not specified by the UNSPSC. The result is that up to 99 different telecommunication services can be categorised with up to 99 different flavours.

The ServiceGrounding of a service (see Figure 5.2) describes how to access the service, more specifically the used communication protocol and message format. Finally the Service Model will be used for the service execution. For this purpose the service is described as a process within the ServiceModel, so a client is able to make use of the service.

Figure 5.3 provides a representation of a service description for the newsticker service mentioned in section 4.5.3. The service description is based on XML. Within the tags "nestedPackage" and "service" the name of the service is specified (here: NewsTicker). These tags are followed by the service operation which is, in this example, "addNewsTicker". The service operation contains two parameters, one as input specified by the ontology class "Video" within the ontology description file "IOTypes.owl" and one as return specified also as an ontology class "Video". Furthermore the service goal is defined and non-functional properties are given (here: "Cost" and "Latency"). This SPATEL-based service description shows that it depends

highly on ontologies and that there is no explanation given how to invoke the service, e.g. a SIP URI is missing.

```
<?xml version="1.0" encoding="UTF -8"?>
<spatel:ServiceLibrary xmi:version="2.0"
 xmlns:xmi="http://www.omg.org/XMI"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:spatel="http://istspice.org/spatel/1.0.0/Spatel"
 name="Service Design">
 <nestedPackage xsi:type="spatel:ServicePackage" name="NewsTicker">
  <service name ="NewsTicker" semPattern="GQIO">
   <ownedOperation xsi:type="spatel:ServiceOperation" name="addNewsTicker">
    <ownedParameter xsi:type="spatel:ServiceParameter"
     name="video" semType="IOTypes.owl:Video" direction="in" instanceType="RTP"/>
    <ownedParameter xsi:type="spatel:ServiceParameter"
     name="video" semType="core.owl:Video" direction="return" instanceType="RTP"/>
    <semTag xmi:id="id" name="OperationGoal" semType="Goals.owl:AddNewsTicker" kind="goal"/>
    <nonFuncTag xmi:id="id" semType="NonFunctional.owl:Cost"
     category="Charging" value="3" isDynamic="false" criterion="Cost"/>
    <nonFuncTag xmi:id="id" semType="NonFunctional.owl:Latency"
     category="QoS" value="2" isDynamic="false" criterion="Latency"/>
   </ownedOperation>
   <semTag xmi:id="id" name="ServiceGoal" semType="Goals.owl:NewsTicker" kind="goal"/>
    <ontology xmi:id="id" name="Goals.owl" uri="http://www.example.com/Goals.owl"/>
    <ontology xmi:id="id" name="IOTypes.owl" uri="http://www.example.com/IOTypes.owl"/>
    <ontology xmi:id="id" name="NonFunctional.owl" uri="http://www.example.com/NonFunctional.owl"/>
    <ontology xmi:id="id" name="core.owl" uri="http://www.example.com/core.owl"/>
  </service>
 </nestedPackage>
</spatel:ServiceLibrary>
```

Figure 5.3: SPATEL service example

Nevertheless the SPICE project already shows the relevance for automated service composition in telecommunications. The presented solution is validated and demonstrates service composition management, it must be highlighted that the usage of web services is not common within the core network of a telecommunication provider. Furthermore the telecommunication provider has to stake a parallel infrastructure to handle the execution of the services triggered by web services (see section 3.4.2). Another point to mention is the usage of ontologies. This might result in complications, because the resulting ontologies may contain ambiguous meanings or the ontology is not precise enough to describe a service and its parameters. As the

results of the evaluations show according to (Euzénat *et al.*, 2009), ontology matching is far away from being fully automated task. In most cases where precision is required, manual intervention will be necessary to verify or fine-tune the matchings produced by the automatic algorithms (Falconer and Noy, 2011), such as service composition algorithms.

The solution given by SPATEL project lacks in preciseness for describing services, for example it is very hard to imagine how communication parameters like the direction of a media stream will be described by such simple ontologies. A very specific and hard to implement characteristic of the SPICE project's ACE is the fact that they use natural language as an input, which can be interpreted in different ways, so the resulting composite services are dependent on the interpretation.

5.1.2 Composition description language for service composition in NGN environments

This language was developed as part of the Multi-Modal Communication and Collaboration Services (MMCCS) project (Bether, 2008). The description language is based on ontologies and syntactically formalisation. The following subsections will describe the concept.

Within a composition, service descriptions are arranged to fulfil a goal. The entity that includes the composition is the composition description, a machine- and human-readable, syntactic and semantic expression of the behaviour of the entire composition and particularly of the interworking services in the composition. A service will be described through metadata, which is optional, one or multiple conditions, which are

runtime parameters, one or multiple policies (agreements between consumers and providers) and one or multiple attributes. Conditions are classified into four types:

- Preconditions: must be considered before the service is processed.
- Post conditions: must be considered after service processing.
- Invariants: must be considered throughout service processing. Is constant through all composition states.
- Rely conditions: Relates to the surrounding system. Influences the system state throughout the composition processing. Based on system parameters.

A composition will be described through metadata, conditions, connectors, transitions, services and attributes. Connectors represent the service flow control. The flow control can be declared by behavioural connectors, which manage transitions between services. The purpose of these connectors is to define the order of the composed member services. Eight basic connectors are defined: start, end, selection, choice, fork, merge, join, transform and loop. Also the flow control is specified by three parameters.

- Input: outcome of the termination stage of the previous service.
- Output: outcome of the processing stage of a service.
- InOut: outcome of the termination stage of the previous service, untouched, or read only, by all stages of that service.

All these parameters can be represented in the syntax formalisation shown in Figure 5.4.

```
Service-Composition(C)
(
        (*identification and dependencies*)
        [METADATA];
        (*global policy agreements*)
        {Policy}-;
        (* global processing constraints*)
        {CONDITIONS
                [INV], [PRE(IN)], [POST(OUT)]
        }-;
        (*global attributes*)
        {ATTRIBUTES}
        CONNECTOR('START'); (*start connector*)
        TRANSITION
        CONNECTOR
                ('SEL' | 'CH' | 'FORK' | 'MERGE' | 'JOIN' |
                 'TRANSFORM' | 'LOOP' | 'ERROR');
        TRANSITION
        SERVICE (*sequence of member services, connectors. transitions*)
                (?...?);
        TRANSITION;
        CONNECTOR('END'); (*end connector*)
)
```

Figure 5.4: Syntax formalisation of service composition in (Bether, 2008)

This can be represented in XML as composition description. The composition consists of an arrangement of a set of behaviour patterns, defining the choreography of a composition. This is called the composition behaviour (see Figure 5.5).

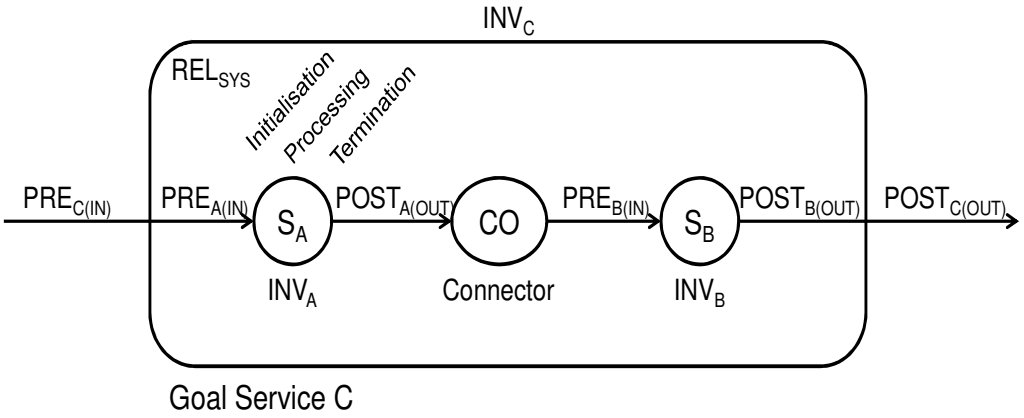


Figure 5.5: Composition behaviour in (Bether, 2008)

The identified behaviour patterns are sequence, parallelism, exclusive decision, multiple decisions and looping.

- Sequence describes a pattern where services are processed exactly in the order in which they were defined, and the output of a caller becomes the input of the following callee.
- Parallelism describes the processing of multiple services at the same time.
- Exclusive decision behaviour describes the case where, out of a set of alternatives, only one is selected and processed further.
- Multiple decision handling is similar to exclusive decision behaviour, but includes multiple paths in the outcome. The decision is based on an optional synchronisation.
- Looping pattern is the construct most commonly used to repeat already defined behaviour.

Furthermore non-functional properties are described. Non-functional properties will describe topics like security, quality, errors and communication forms (e.g. synchronous or asynchronous communication).

Also a telecommunication service ontology is introduced by (Bether, 2008), which is represented in Figure 5.6. The service ontology is described by three semantic perspectives: META, STATIC and DYNAMIC. The META semantics help to identify and describe a service cluster. The META semantics keeps three clauses, where the ID and Name represent unique identifiers and the Dependencies describes the cohesion between the clusters, coarsely assessed by the qualities of weak and strong (see Figure 5.7).

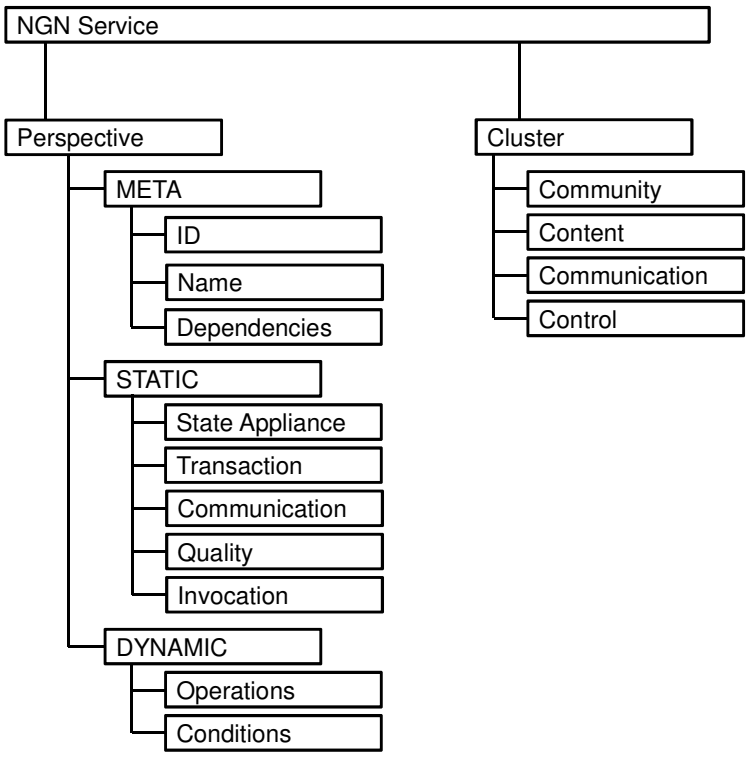


Figure 5.6: Telecommunication service ontology in (Bether, 2008)

The STATIC semantics describe a service cluster in its non-composed atomic state, providing useful information about the service before it is processed. And the DYNAMIC semantics interpret service cluster capabilities, which are important in the behaviour of a cluster (e.g., during composition). The service ontology also contains information about the four interacting service clusters: community, content, communication and control services. These clusters aggregate similar service functionality and interact among themselves. The four service clusters are specified as follows.

Community Services

The functionality of community services are human consumer-centric and provide shared, largely non-technological, common information. They are driven by human interest. To identify such services they are provided with a special META semantic ID

and name. Community services are strongly bound to other service clusters such as content, communication and control. For example, online communities involve rich interactions between users based on content (e.g. images and videos), communication (e.g. SMS messaging), and control (e.g. central session handling). STATIC semantics are for example the response time, availability and scalability of the service. Also security and trusted access are required attributes. The DYNAMIC semantics are for example conditions belonging to quality aspects such as the availability of the service.

Content Services

The content cluster represents data-centric services, for instance a media streaming service such as Internet television. Content is a digitalisation of information as media. Content services do not necessarily rely on community services. Further they do not necessarily cooperate with communication services. Content does cooperate mandatorily with control services (e.g., controlling the direction of delivery).

Communication Services

A communication service is for instance email or a telephone call. Communication services represent interaction-centric functionality and consist of single or multiple interactions. They control the connectivity between human users to guarantee a reliable transmission of information. Communication and control services are strongly bounded. This implies the explicit handling of signalling and delivery of conversational data, especially when users or machines interact in real time. Different aspects will modify the STATIC semantics. These are for example real-time and non-real-time interactions, simultaneous media delivery (e.g. text, audio and video), bandwidth, packet delay, and packet jitter and packet loss. Also media streams can be

characterised by different modes (e.g. pull and push services). The DYNAMIC semantics are specified by functionalities of signalling and delivery.

Control Services

A control service for example is that which enables the starting, stopping or pausing a media stream. Control services act as a central authority, taking responsibility for controlling user and media data across all other service clusters. The strong dependencies between control services and each of the other clusters, especially communities (as regards user control) and communication (real time media control) were already mentioned. The descriptive DYNAMIC semantics are user control, media control and session control.

The following Figure 5.7 summarises the ontology, illustrating the clusters, their semantic clauses and the cohesion between them.

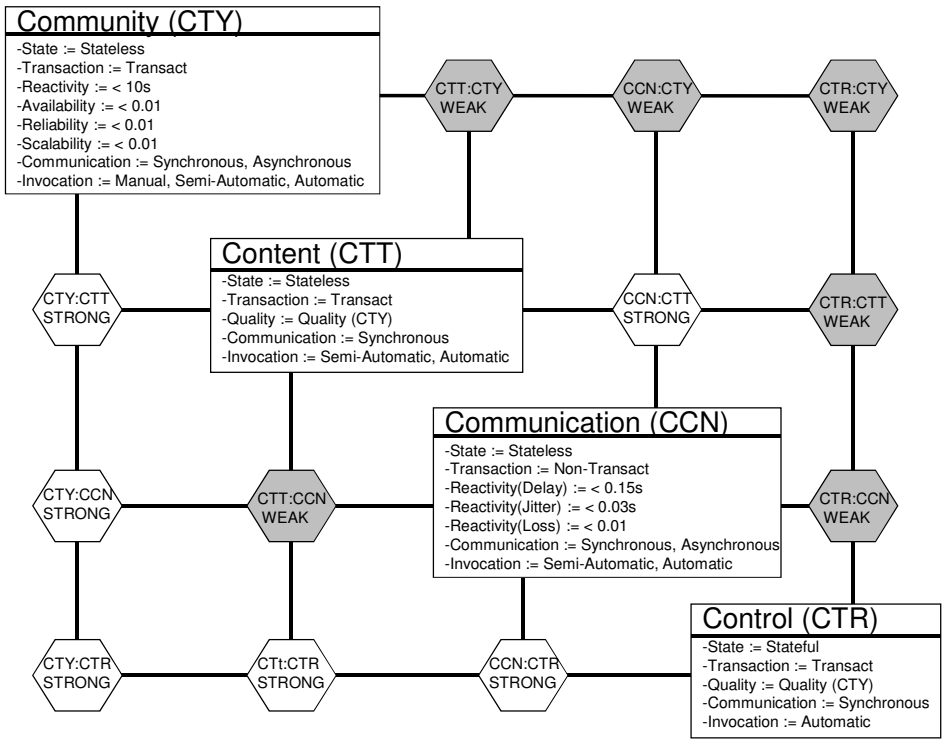


Figure 5.7: Service cluster – cohesion and clauses (Bether, 2008)

In Figure 5.8 and Figure 5.9 the newsticker service from section 4.5.3 is represented by the here specified approach of a service description language. The first thing to mention is that the service description consists of two different descriptions. One is based on WSDL (Web Service Description Language) (see Figure 5.8) and the other is a simple XML-based document (see Figure 5.9). Within the WSDL document the information on how to access the service is given. The SIP URI is defined under the "wsdl:port" tag and the SIP method is defined under the tag "wsdl:binding". This document contains not enough information about the capabilities provided by the service, which means, that the WSDL description is not sufficient to describe all properties of the service.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:schema="http://example.com/schemas"
  xmlns:tns="http://example.com/services" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  targetNamespace="http://example.com/services">
<wsdl:types>
 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:import namespace="http://example.com/schemas" schemaLocation="SIPservice.xsd"/>
 </xsd:schema>
</wsdl:types>
<wsdl:message name="NewsTickerRequest">
 <wsdl:part name="siprequest" element="schema:Request"/>
</wsdl:message>
<wsdl:portType name="NewsTickerPortType">
 <wsdl:operation name="NewsTicker">
  <wsdl:input message="tns:NewsTickerRequest"/>
 </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="NewsTickerBinding" type="tns:NewsTickerPortType">
  <binding verb="INVITE"/>
   <wsdl:operation name="NewsTicker">
    <operation location="urn:/#NewsTicker"/>
     <wsdl:input name="NewsTickerRequest"/>
    </wsdl:operation>
  </wsdl:binding>
<wsdl:service name="NewsTickerService">
<wsdl:port name="NewsTickerPort" binding="tns:NewsTickerBinding">
  <address location="sip:newsticker.bbc@bt.co.uk"/>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

Figure 5.8: WSDL part of sample service

The second part of the service description contains the meta properties: state, transaction, quality and communication and invocation, expanded in Figure 5.9. The newsticker service is a communication service, so it should have the properties state ”Stateless”, transaction ”Non-Transaction”, quality ”Reactivity<0.15s”, communication ”Synchronous or Asynchronous” and invocation ”Semi-Automatic or Automatic”. Not all of the properties are illustrated within the Figure 5.9, because it does not contribute to further understanding. The transaction support is set to false, the reactivity is set to less than 1 second and the communication is set to synchronous.

```
<?xml version="1.0" encoding="UTF-8"?>
<al:Properties Type="MetaProperties" Identifier="200:210:212"  xmlns:al="http://example.com/schemas"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://example.com/schemas/NGNServicePropertyTypes.xsd">
<al:StateManagement>
  <al:SessionSupport>
    <al:Name>IsEnabled</al:Name>
    <al:Value>TRUE</al:Value>
  </al:SessionSupport>
</al:StateManagement>
<al:TransactionHandling>
  <al:TransactionSupport>
    <al:Name>IsEnabled</al:Name>
    <al:Value>FALSE</al:Value>
  </al:TransactionSupport>
</al:TransactionHandling>
<al:Quality>
  <al:ReactivityMetric>
    <al:Name>ReactivityMetric</al:Name>
    <al:Conditions>
      <al:Name>ReactivityMetricCondition</al:Name>
      <al:Restriction>
        <al:Type>Arithmetic</al:Type>
        <al:Restriction>Less</al:Restriction>
        <al:Threshold>1</al:Threshold>
        <al:Unit>Seconds</al:Unit>
      </al:Restriction>
    </al:Conditions>
  </al:ReactivityMetric>
</al:Quality>
<al:Communication>
  <al:Mode>
    <al:Name>Synchronous</al:Name>
    <al:Value>TRUE</al:Value>
  </al:Mode>
</al:Communication>
</al:Properties>
```

Figure 5.9: Meta properties for sample service

The presented concept of (Bether, 2008) is well elaborated from the view of a provider and better formulated than the one presented in SPICE project. The used ontologies and service clusters can be used to specify services, but some properties are missing, for instance the directionality of communications (unidirectional as a broadcasting service or bi-directional like a telephone call) or the intrinsic time dependency. The term intrinsic time dependency is interpreted as that the information which is transported within a communication has a time dependency, for example video, audio, and animation are time-based information, in contrast to non-time-based information, which includes for instance images, graphics and text. Furthermore, the number of parties regarding to a communication service is a missing property, for instance it is important to know how many parties are supported by a conferencing service. Without the support of these properties, an automated service composition will not work for every value-added service. The description language should be more precisely formulated.

5.1.3 A new approach to classify and describe telecommunication services

In (Lehmann *et al.,* 2009b) a novel approach to describe services is presented. This approach tries to describe services from a new point of view. Services in telecommunications always offer different media. Media are classified according to media perception by humans and media representation.

Media perception describes the modality of how humans will conceive information. The information humans may conceive is linked to their sensory perception. Therefore only the physiological perception is considered and no subjective or other natured

scoring. Examples of such perceptible media are acoustic (music, speech and sound) and visual (text, still image and video) media. The representational medium belongs to the processing of information e.g. the encoding of video with MPEG-format (Moving Pictures Experts Group) (Boles *et al.,* 1996).

A media type will be described through its representation and perception of media. It will be named by its media perception (e.g. audio, video, text). Media types can be time-invariant or continuous. Time-invariant media types are for example text and still image and continuous media types are for example audio, video, animation (Boles *et al.,* 1996).

The data of multimedia can be processed by the following components (Boles *et al.,* 1996):

- Sources output data streams of direct media types.
- Sinks receive data streams and they can potentially display them.
- Filters combine the characteristics of sources and sinks. Filters are mostly used to convert data streams (e.g. mono-to-stereo).

These three components are called media objects. All media objects are connected by media channels/ports that belong to one media type. These ports represent the I/O-interfaces of a media object. Figure 5.10 shows an overview of the media objects.

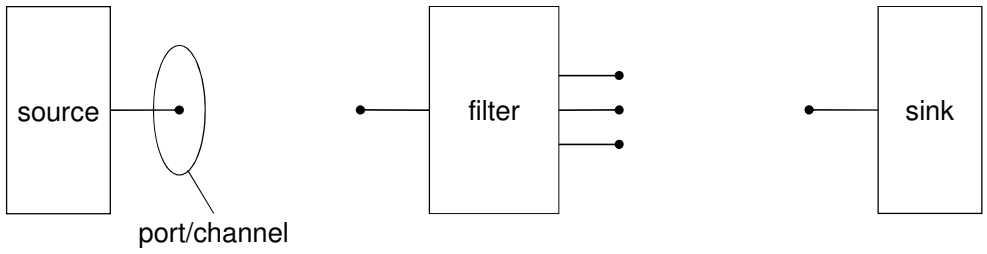


Figure 5.10: Media objects

These elements are well known from telecommunications. Telecommunication services will support more media types, which can be derived from human senses (Lehmann *et al.,* 2009b). All five senses are shown in the following listing.

- Audible (Hearing)
- Visual (Sight)
- Gustatory (Taste)
- Olfactory (Smell)
- Haptic (Touch)

Four further senses (pain, balance, proprioception and temperature) are known, which will be assigned to the sense touch, only for simplification. Furthermore there are also media types which are not sensually imperceptible. These media types can only be processed by machines. Following they will simply be called data. Consequently there are six different media types. They could be characterised more precisely (e.g. Hearing: Audio, Speech or Sound). Table 5.2 gives an overview of the media types and some of their possible characteristics.

Table 5.2: Media types and their characteristics

Media type	**Perceptual Characteristics**
Hearing	sound, speech, music
Sight	video, animation, text, still image, light
Taste	sweet, sour, bitter, spicy, umami
Smell	camphorous, fishy, malty, minty, musky, spermous, sweaty, urinous
Touch	pressure, temperature, balance, proprioceptive
Data	file, sensor, actuator, trigger, information

The shown classification of media objects and types creates a possibility to describe compositions of multimedia streams. Media compositions are divided into the three types: spatial, temporal and configurable compositions (Steinmetz, 2000). Spatial composition describes the geometrical structure of the presentation of the media (e.g. the position and size of a video within a frame of a web site) and will not describe the way a service should work. The configurable composition describes dynamic connections between the single media objects (e.g. how text has to be converted into speech) and the temporal composition describes temporal relations of the media objects. The relevant media composition for telecommunication services is the temporal composition combined with a configurable composition called service function. According to (Allen, 1983) relations between two intervals can be divided into 13 different relations. Figure 5.11 shows seven of them. All other relations result in inverted relations. The only relation without inversion is "equals".

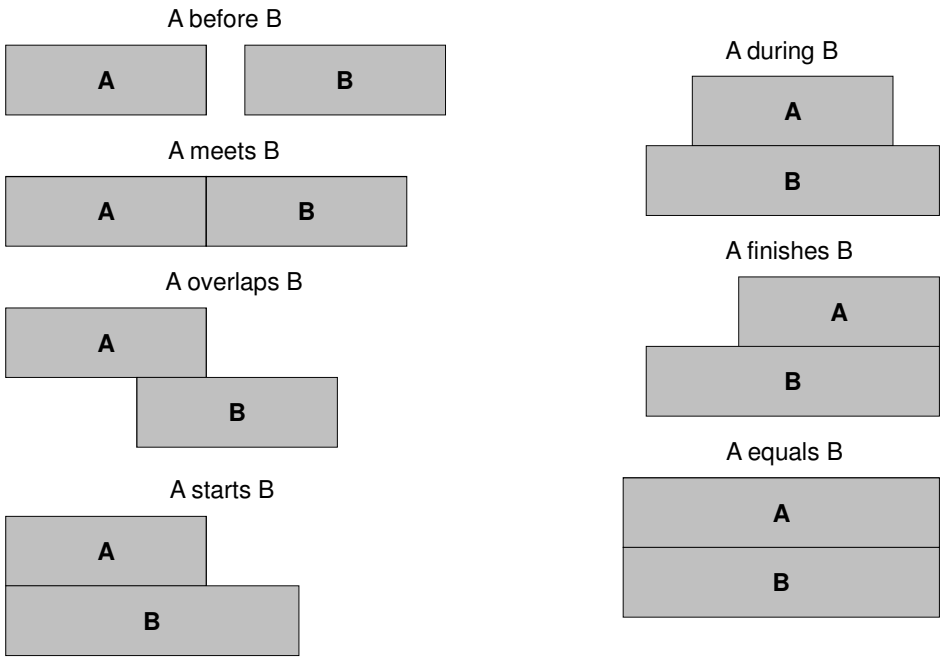


Figure 5.11: The seven possible relationships between intervals (according to Allen, 1983)

With the presented methods it is still possible to describe multimedia services. But this description is very vague. To specify the description more precisely the media objects are extended with inner methods. The inner methods are derived from the possible basic connection types in telecommunications (see Figure 5.12).

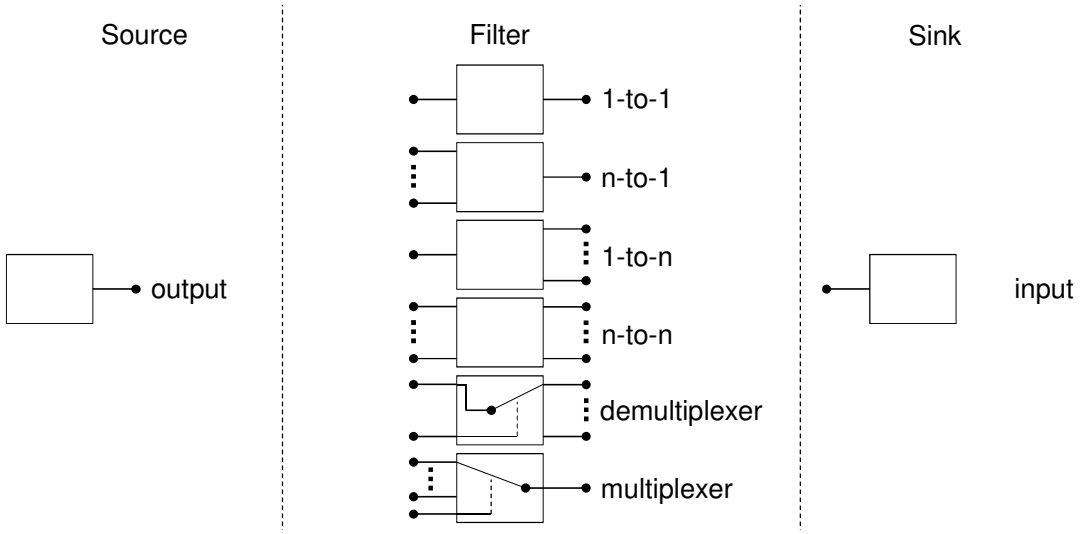


Figure 5.12: Basic types of media objects

The shown basic types of the filter elements are sorted into three groups, intra-media, inter-media and special types. The intra-media filters will only process one media type, inter-media filters will handle different media types, while the special types can be inter or intra media. Figure 5.13 presents an overview of the different filter types.

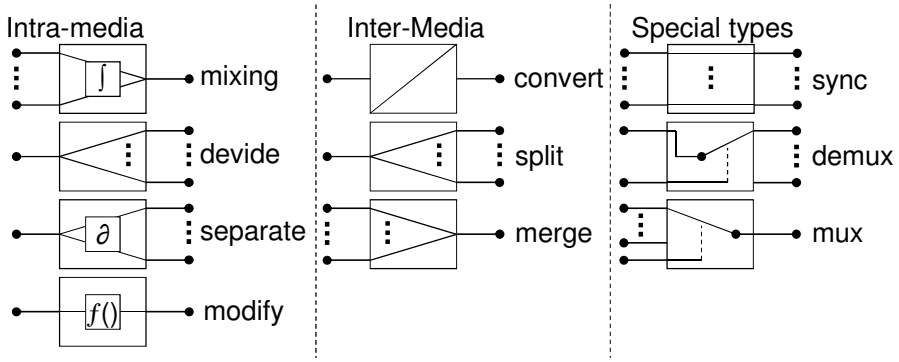


Figure 5.13: Filter types

The following example demonstrates how to describe telecommunication services by using the defined elements and methods mentioned above. The example describes a simple service scenario.

Some information (e.g. the identity of a subscriber) will be displayed to another subscriber; if the information matches to a simple result of a function (e.g. the initiating subscriber's name is "bob"). Afterwards the two subscribers may have a multimedia session to communicate with each other (e.g. audio and video telephony).

To describe the service abstracted by the usage of the elements and methods presented above it is essential to distinguish two different types of descriptions. The first one is the service function description and the second is the temporal service composition. Both are shown in Figure 5.14.

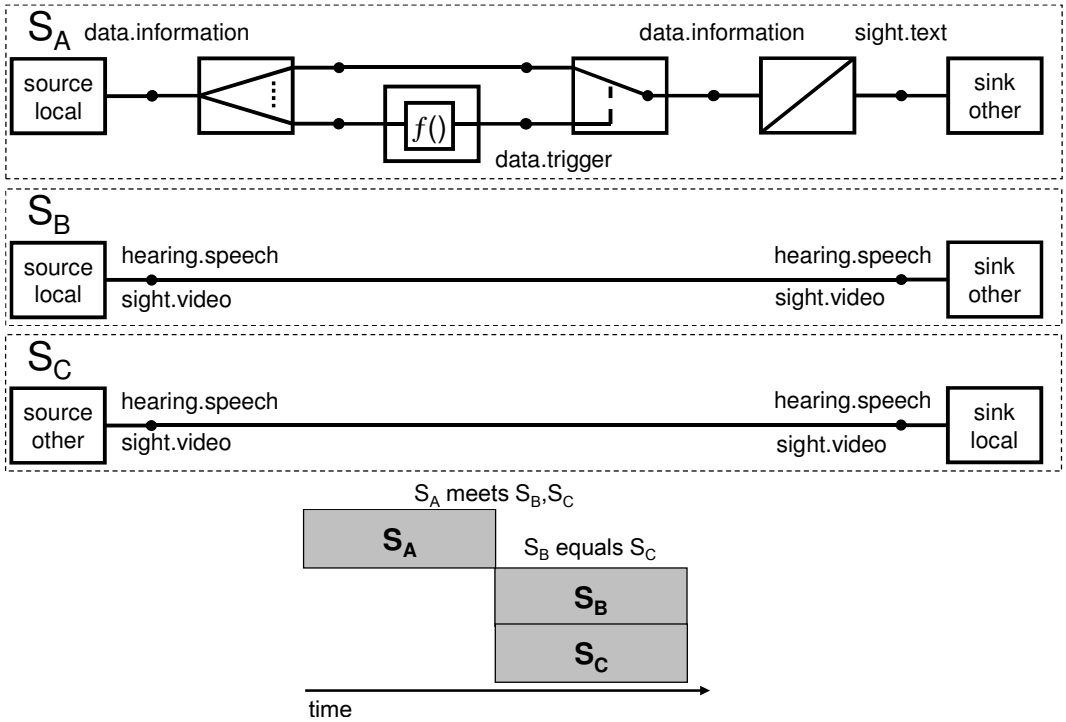


Figure 5.14: Exemplary service description using approach to classify and describe services

The service function description in Figure 5.14 depicts three different service functionalities. The first one, "S_A", is the functionality that if the transmitted information containing the wanted data, such as “bob”, it triggers the media stream and converts the data to transmit. The descriptions "S_B" and "S_C" introduce two media streams, namely speech and video. All three functional descriptions are temporarily composed.

The newsticker service is illustrated in Figure 5.15 for a better comparison with the previously discussed two service description languages (see sections 5.1.1 and 5.1.2). As the representation of the newsticker’s service description displays, there is no information about the signalling protocol even no specified SIP URI and furthermore the functionality of the service is not really described. The filter type *modify* (see Figure 5.13) simply defines in conjunction with the given information regarding the media type *sight.video*, that media will be modified internally. Nothing indicates that the function of the filter type integrates the functionality of a newsticker service.

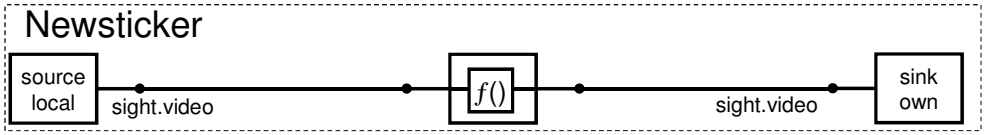


Figure 5.15: Newsticker service using (Lehmann *et al.,* 2009b)

This section presents a novel, conceptually different approach to describe telecommunication services, based on describing services from a media view. The approach is incomplete because it does not specify preconditions or postconditions, as well as non-functional properties (e.g. security).

5.2 Proposed SDL for service composition

As highlighted in sections 5.1.1, 5.1.2 and 5.1.3, the notations introduced are incomplete in exhaustively specifying a service, and it requires an SDL to fulfil the completeness requirement. The aim of the proposed SDL is that it fulfils all the needed requirements, which are listed here:

- Machine readable/interpretable – by this service compositions can be created automatically.
- Interface description – it contains information such as URI and protocol to make use of the service.
- Unique identifier – by this a service can be discovered by the discovery server.
- Independency – the independency of platform and programming language, which is derived from the SOA.
- Functional properties – which are used to identify the service features and the In- and Outputs to form compositions.
- Non-functional properties – to characterise and rank the services, for instance by their cost or QoS.
- Usability – the usability by developers and end-users/subscribers is essential.
- Standardised language – the SDL must be based on standardised language(s) (e.g. XML, OWL-S, web services).
- Preciseness – the unambiguity of SDL crucial.

This section proposes a service description language that, consists of five constructs: *service Id*, *temporal composition*, *service goal*, *functional properties* and *non-functional properties*. The *service Id* is an identifier for a non-composite service, which

will be named atomic service, represented by its permanent SIP URI (e.g. sip: service1.as1@bt.co.uk). The composition of atomic services will be named molecular service. The *service goal* is a set of keywords describing the functionality so that humans can infer what the service will offer. Beside the keywords, an SDL can include an optional description to fully describe the service functionality. For example a service goal with the keywords "wake up" and "instant message" might describe a service that will send an instant message as a wake up message.

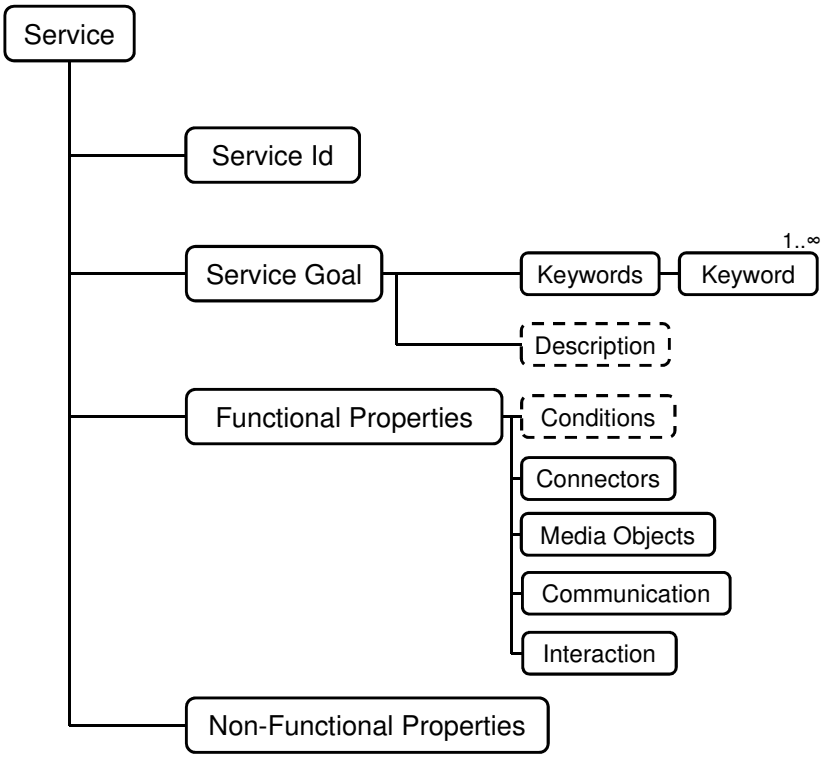


Figure 5.16: Structure of the service description language

Non-functional properties, such as cost or security, are used to limit the subset of compositions that fulfil the service request and rank the generated set of compositions, such as ranking higher a service with lower costs and higher reliability.

The *functional properties* are the *conditions* (pre-, post-condition) which are optional, because there might be services without any conditions, *connectors* (in-, through- and output), *media objects* (source, sink and filter), *communication* (asynchronous/synchronous and form of communication) and *interaction* (non-real-time/real-time). The *conditions* are properties that must be considered before or after service processing; an example of a condition is that, prior to any interaction, the SIP session must be established. The *connectors* are major elements for aggregating services. All in-, out- and throughputs are offering attributes to identify if the *connectors* are for the media stream of the originating party (the party who initiated the service invocation) or if they belong to the terminating party. *Connectors* also may be optional, this means they can be utilised if they are needed, for example in conferences there might be connectors for participants that will join at a later point. Furthermore, the *connectors* are linked to *media types* (see Figure 5.17). These *media types* are derived from human senses (hearing, sight, smell, touch, taste) and a non-sensory *media type* (see section 5.1). Four further senses (pain, balance, proprioception and temperature) are known, which will be assigned to the sense touch, only for simplification. These *media types* will cover a broad range of transferable media in telecommunication.

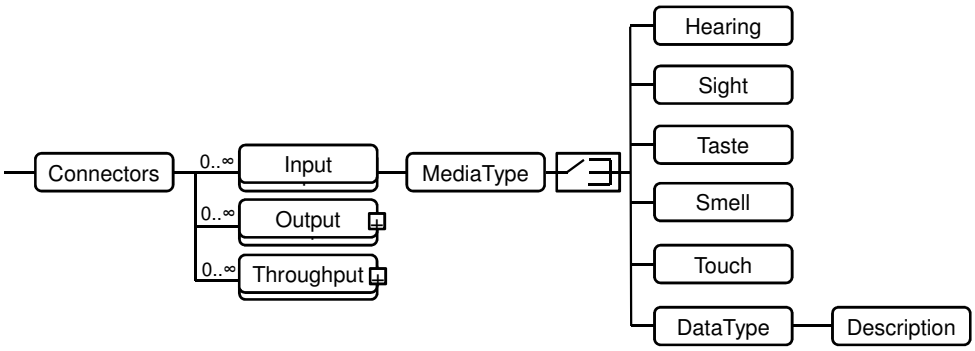


Figure 5.17: Structure of the connectors

The *connectors* are forming ports for linking them with the matching opposite ones. For example out- and throughputs can only be linked to in- or throughputs which will fit, that is they have to be from the same *media type* and must own the same characteristic (such as Hearing and Sound). Ports of different *media types* and different characteristics are incompatible and therefore cannot be combined. The characteristics (shown in Table 5.2) are specifications of the *media types*, so they can be assigned more precisely. This is essential in communication networks, because different media type characteristics may have different bandwidths/sampling rates; for example, standard sampling rate for speech is 8 kHz in contrast to music in CD-quality, which is 44.1 kHz. In telephony special codecs exists for these different demands. The grading of quality aspects within one characteristic, such as speech, is a *non-functional property*. In contrast, the imperceptible media type is characterised by different technical aspects or formats. This is required for aggregating the same characteristics of imperceptible *media types*.

The *media objects* constructs include media *sources*, *sinks* or *filters* (see Figure 5.18). The *sources* output data streams and the *sinks* receive data streams of specific, predefined media types. *Filters* combine the characteristics of *sources* and *sinks* and

they are mostly used to manipulate data streams. For this purpose, the *filters* are divided into different functionalities namely *input/output-selector*, *synchroniser* (synchronise different *media types* such as speech and video), *inter-media* and *intra-media*. The *input-selector* is a device that selects one of several input *media types* and forwards the selected *media type* as output. On the other hand an *output-selector* takes a single input *media type* and forwards it to a chosen output from several. Both selector objects are controlled by an input *media type* (e.g. an imperceptible media input like sensor controls the *media types* with the characteristic hearing). *Inter-media filters* are processing different *media types*. However *intra-media filters* are only processing one *media type*.

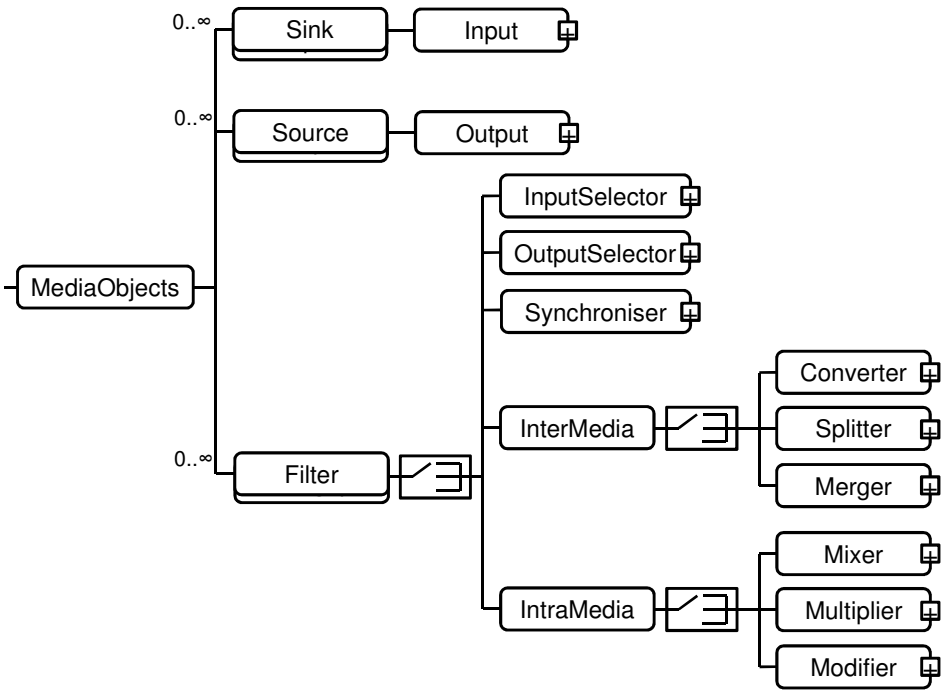


Figure 5.18: Media objects structure

These two media *filters* can be divided into *converters* (for example Text-to-Speech), *splitters* (for example separate subtitle from video) and *mergers* (for example bundle speech and video to a stream as output). The *media objects*, *inter-media filters*, and

intra-media filters can be divided into *mixer* (for example video conferences or text chat), *multiplier* (duplicate the media type), and *modifier* (for example change the size of a video).

General information must be defined for all *media objects,* this is the form of *communication* (for example one-to-one, many-to-one) and last but not least the way of *interaction* has to be defined. For example, a real-time service will be a normal voice call in contrast to instant messaging, which is a non-real-time service.

Figure 5.19 illustrates a service description for a service identifiable by its keyword "recognition", represented in XML. As depicted the *connectors* input is of the *media type* **hearing** with the characteristic **speech** and the output is of the *media type* **sight** with the characteristic **text**, so the service is a speech recognition service which will return the result as text information.

```
<?xml version="1.0" encoding="UTF-8"?>
<Service xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="../ServiceDescription.xsd">
        <ServiceId id="sip:recogniser.as1@bt.co.uk"/>
                <ServiceGoal>
                        <Keywords>
                                <Keyword>recognition</Keyword>
                        </Keywords>
                </ServiceGoal>
        <FunctionalProperties>
                <Connectors>
                        <Input sessioncase="originating" id="1">
                                <MediaType><Hearing audition="speech"/></MediaType>
                        </Input>
                        <Output sessioncase="originating" id="1">
                                <MediaType><Sight vision="text"/></MediaType>
                        </Output>
                </Connectors>
                <MediaObjects>
                        <Filter>
                                <InterMedia>
                                        <Converter>
                                                <Inputs><InputId id="1"/></Inputs>
                                                <OutputId id="1"/>
                                        </Converter>
                                </InterMedia>
                        </Filter>
                </MediaObjects>
                <Communication>
                        <Synchronicity type="asynch"/>
                        <form><One-One/></form>
                </Communication>
                <Interaction reaction="real-time"/>
        </FunctionalProperties>
        <Non-FunctionalProperties>
                <Quality><Content content-qos="high"/></Quality>
        </Non-FunctionalProperties>
</Service>
```

Figure 5.19: Service description example

The <MediaObjects> XML tags allow a more accurate definition of the used *media object* as an *inter-media filter* of the type *converter*.

The utilisation of the proposed SDL can be illustrated using the service scenario mentioned in section 4.5.3. This scenario is a service combination out of an audio conference and a video conference with an embedded news ticker (e.g., actual football scores). First it is necessary to take a look at the service descriptions of the three services (audio conference, video conference and news ticker). These services will be discussed in the following subsections based on their *functional properties*. The *service Ids* are audioconference.telekom@bt.co.uk for the audio conference service, videoconference.bt@bt.co.uk for video conference service and

newsticker.bbc@bt.co.uk for the final service. The properties of the audio conference service are shown in Figure 5.20.

```
<Service>
    <ServiceId id="sip:audioconference.telekom@bt.co.uk"/>
        <ServiceGoal>
            <Keywords>
                <Keyword>conference</Keyword>
            </Keywords>
        </ServiceGoal>
    <FunctionalProperties>
        <Connectors>
            <Input sessioncase="originating" id="1">
                <MediaType><Hearing audition="speech"/></MediaType>
            </Input>
            … further inputs
            <Output sessioncase="originating" id="1">
                <MediaType><Hearing audition="speech"/></MediaType>
            </Output>
            … further outputs
        </Connectors>
        <MediaObjects>
            <Filter>
                <IntraMedia>
                    <Mixer>
                        <Inputs>1,2,…,m</Inputs>
                        <Outputs>1,2,…,n</Outputs>
                    </Mixer>
                </IntraMedia>
            </Filter>
        </MediaObjects>
        <Communication>
            <form><Many-Many/></form>
        </Communication>
        <Interaction reaction="real-time"/>
    </FunctionalProperties>
</Service>
```

Figure 5.20: Audio conference service description

The only differences between the description of the video and the audio conference services are in the *connectors* and the *serviceID*. The *media type* of the audio conference service is **hearing** and the *media type* for the video conference service is **sight** with characteristic video. The properties of the news ticker service are presented in Figure 5.21.

```
<Service>
    <ServiceId id="sip:newsticker.bbc@bt.co.uk"/>
        <ServiceGoal>
            <Keywords>
                <Keyword>information</Keyword>
            </Keywords>
        </ServiceGoal>
    <FunctionalProperties>
        <Connectors>
            <Input sessioncase="originating" id="1">
                <MediaType><Sight vision="video"/></MediaType>
            </Input>
            <Output sessioncase="originating" id="1">
                <MediaType><Sight vision="video"/></MediaType>
            </Output>
        </Connectors>
        <MediaObjects>
            <Filter>
                <IntraMedia>
                    <Modifier>
                        <Inputs>1</Inputs>
                        <Outputs>1</Outputs>
                    </Modifier>
                </IntraMedia>
            </Filter>
        </MediaObjects>
        <Communication>
            <form><One-One/></form>
        </Communication>
        <Interaction reaction="real-time"/>
    </FunctionalProperties>
</Service>
```

Figure 5.21: News ticker service description

The invocation of the service composition as presented in section 4.5.3 is done by utilisation of SIP and the Route header, thus the *serviceIDs* defined in the service descriptions are used. Figure 5.22 gives an overview of the resulting media streams and the services. The handling of the media streams depends on the service descriptions inputs and outputs. As clearly illustrated in Figure 5.22 the attributes originating/terminating in connection to the inputs and outputs are belonging to the originating site respectively the terminating site. The service composition will be built by the algorithm, which is presented in section 5.4.

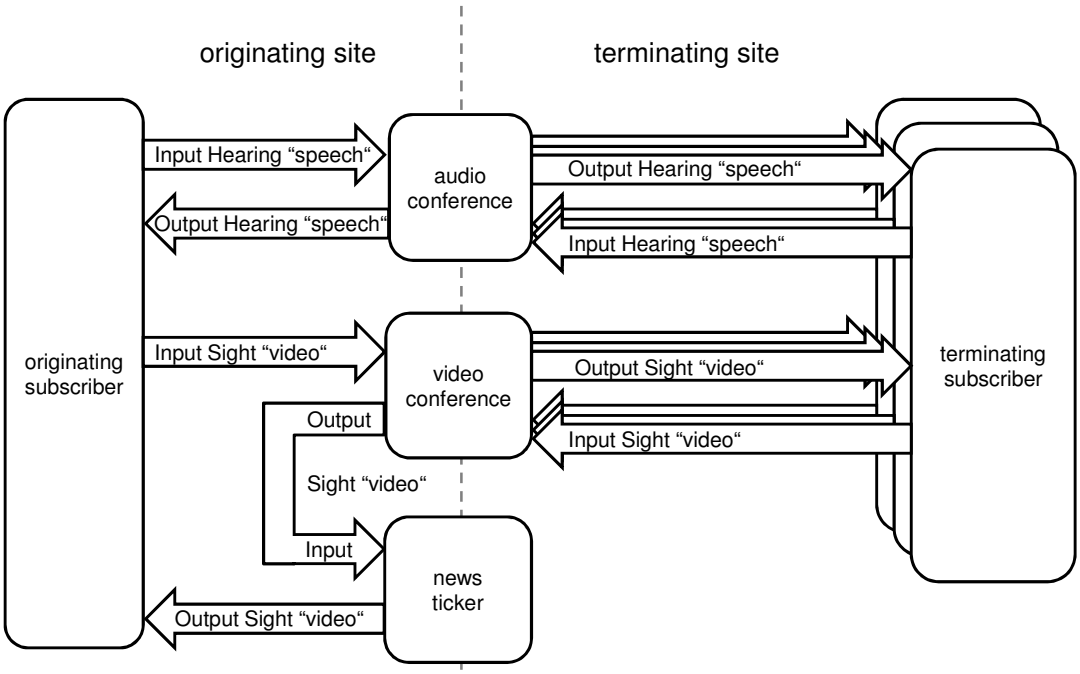


Figure 5.22: Service composition media streams

The presented service composition had to be created out of the three different service descriptions by the discovery server and composition server as described in section 4.3 and 4.4.

Furthermore the service composition request procedure has to be depicted. As described in chapter 4, service compositions are triggered by service requests. The request for a service composition is formed by a sequence of keywords which are linked by the pipe symbol | (see section 4.4), to specify that these services will exchange media. This is derived from the well-known Unix pipes used for pipeline of commands (Ritchie, 1984); here it is used for the pipeline of services represented by keywords, for example *videoconference|newsticker*. The piping allows defining a chain of services whereby the data flow between the services is specified. This means the data flows always from the first element to the next and so on.

Two further elements are parts of the syntax used to describe service compositions on basis of keywords. The first are the round brackets *( )* which will be used to form groups of service chains, e.g. *(videoconference|newsticker) (audioconference)*. The second element is a unique identifier, introduced as <*name*>, which can be set to specify the reutilisation of the services. A side effect is that the service needs to have more inputs respectively outputs or throughputs. A short example would be *(videoconference.id_x|newsticker) (id_x|recorder)*, where the videoconference service is reused, one output is connected to the input of the newsticker service and another output is connected to a recorder service. This simple syntax allows specifying relatively complex service compositions, potentially by subscribers too.

The proposed service description language fulfils the needed requirements that are not supported by the previously described languages. This proposed SDL is not ontology based, so the keywords should be unambiguous, relying on a strict and precise definition for every keyword, which would fall under the responsibility of the core infrastructure provider (operator of the frameworks core elements). An advantage of the proposed SDL is the independence of protocols and underlying technologies; for instance web services can be used interchangeably with any other technologies. An additional advantage is that the SDL has an XML-based structure, which makes it platform and programming language independent. Further mentioned requirements are covered, such as the wide range of definable functional properties. The functional properties are defining the directionality of communication (e.g. originating or terminating), the number of supported parties (e.g. participants supported by a conference room), the potential conditions (e.g. preconditions and/or postconditions) and the capability to use well-defined *media types* and *media objects* to describe the

functionality of a service (e.g. a mixer for speech). Non-functional aspects can also be specified, such as security aspects or further information about the anticipated QoS. An additional advantage of the illustrated SDL is that only one document is necessary (see SPATEL and Composition description language for service composition in NGN environments).

5.3 Comparison of SDLs

Following from their introduction, this section compares the service description languages from sections 5.1 and 5.2 to illustrate their relative strengths and weaknesses. For this purpose different characteristics are chosen. The following Table 5.3 employs a three level scale, using (-) for absence of feature or functionality, (o) for basic support of feature or functionality, and (+) to represent advanced support of the feature or functionality.

Several characteristics for SDLs are derived from the SOA principles (see section 3.1), the platform and technology neutral service interfaces (interface description and independency of platform and programing language) and the service discoverability (detectable by a unique identifier). Furthermore, the SDL has to be understandable by a machine, allowing it to interpret the SDL and build compositions. Also some requirements, the support of functional and non-functional properties, are derived from other SDLs like SPATEL. To enable developers as well as end-users to request or create service compositions the SDL should be described from end-users view. To foster openness, reusability and interoperability the SDL should be based on standardised language(s). As already mentioned within the section 5.1 the unambiguity of the SDL is a must.

The following characteristics are based on the results of the previous chapters (see sections 3.5 and 4.6) and the service class specified in (Martin *et al*., 2004). The different characteristics of the SDLs to match against each other are:

- Readability/Interpretability by machine.
- Interface description (e.g. URI, protocol, etc.).
- Unique identifier of a service.
- Independency of platform and programming language.
- Support of functional properties (e.g. Service Goal, In/Outputs, Conditions).
- Support of non-functional properties (e.g. costs, QoS).
- Usability by developers and end-users/subscribers (described from end-users view not providers view).
- Based on standardised language(s) (e.g. XML, OWL-S, web services).
- Preciseness respectively unambiguity of SDL.

Further characteristics such as generalisation, conciseness and efficiency are not within the scope of this research work. Also ontology is not presented as a characteristic, because ontology might bring ambiguity regarding keywords to classify services. As a result ontology is not a solution for a service description language to fulfil the needs in preciseness.

Table 5.3: Comparison of Service Description Languages

Characteristics	Service Description Language			
	SPATEL	Composition description language	Approach to classify and describe services	Proposed SDL
Readability/Interpretability	+	+	-	+
Interface description	+	+	-	+
Unique identifier	+	+	-	+
Independency	+	+	+	+
Functional properties	+	+	+	+
Non-functional properties	+	+	-	+
Subscribers usability	+	-	o	+
Based on standardised language(s)	+ (OWL-S, web services)	+ (XML, web services)	-	+ (XML)
preciseness	-	o	o	+

All presented SDLs are interpretable by machines except the approach to classify and describe services. Also an interface description is missing in approach to classify and describe services; however the other SDL's provide the description of interfaces to make usage of the services. Even a unique service identifier is not present in the approach to classify and describe services, which prevents the usage of services with the same functionality. The possibility to describe non-functional properties might be obligatory for instance to describe QoS parameters. One essential feature of a SDL should be the applicability by subscribers, because they will request for services and service compositions. Another crucial characteristic is that the SDL has to be based on a standardised language, so that it is reusable and interpretable by everyone. The decisive characteristic here is the preciseness of the SDL. SPATEL is not that precise, because it has no possibility to describe communication parameters e.g. the directionality of media stream, also the usage of ontologies might lead to

misinterpretations. The composition description language and approach to classify and describe services are not precise enough, because they do not have the ability to define communication parameters like the directionality of media or number of parties. As a result of the Table 5.3 the proposed SDL supports all characteristics, so it is chosen for the proposed framework.

5.4 Algorithm for service composition

The algorithm for composing services is developed based on the presented syntax from section 5.2. In the overall processing of the algorithm will be described. In Figure 5.23 a simplified flow chart is presented.

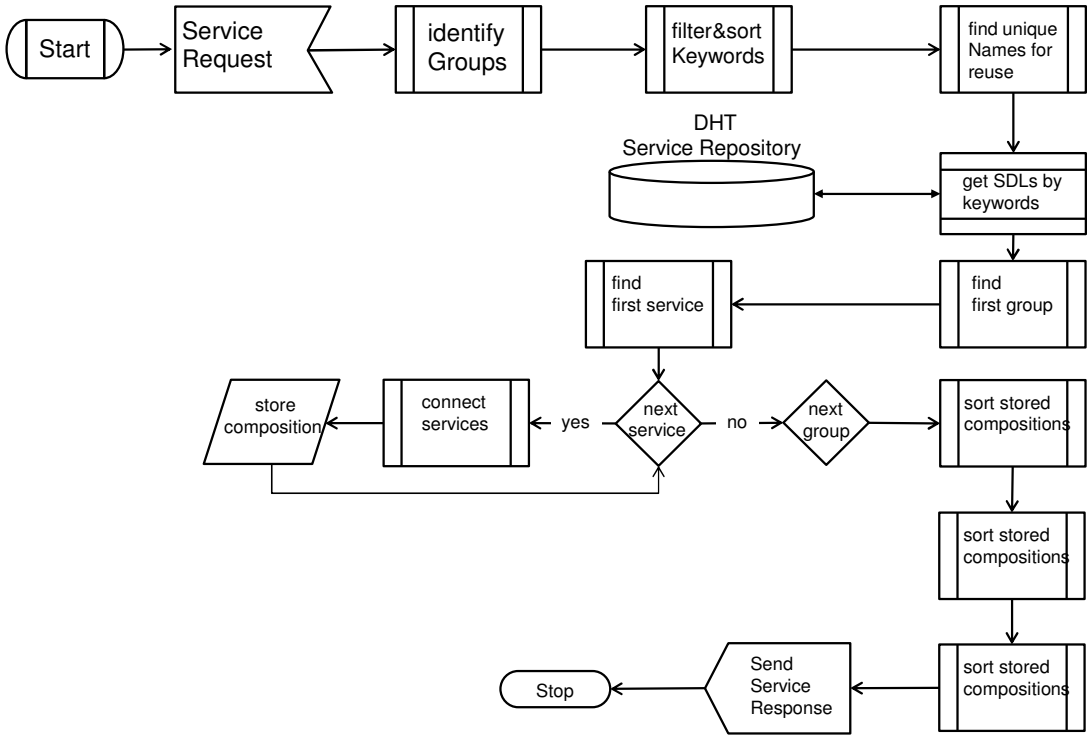


Figure 5.23: Service composition algorithm flow chart

When the request for a service composition is received the groups within the request are identified (e.g. *(videoconference | newsticker)* ≡ *groupA* and *(audioconference)* ≡

groupB), if they exist. After that, the keywords of each group are detected and sorted by their appearance. If keywords are connected to unique identifiers, they will be detected and then the service descriptions related to the keywords are requested from the DHT-based service repository. In the next step the first identified group, e.g. *groupA*, is chosen. Within this group the first service is taken, here: *videoconference*.

Each service is specified by its *connectors* and can be described as a graph (Kalasapur *et al.* 2007). The graph $G_S=\{V_s, E_s, \beta_s, \varepsilon_s\}$ is a 4-tuple which consists of V_s representing a service itself, E_s a set of directed edges which are representing the inputs, outputs and throughputs of the service, and the attribute functions β_s and ε_s describing the vertices and edges. The attribute function β_s adds the attributes to the service, e.g. *service Id*, *service goal*, and *non-functional properties*. The edge attributes, including the *functional properties*, e.g. *conditions*, *connectors* and *media objects*, are represented by the function ε_s. Figure 5.24 illustrates the representation of a videoconference service.

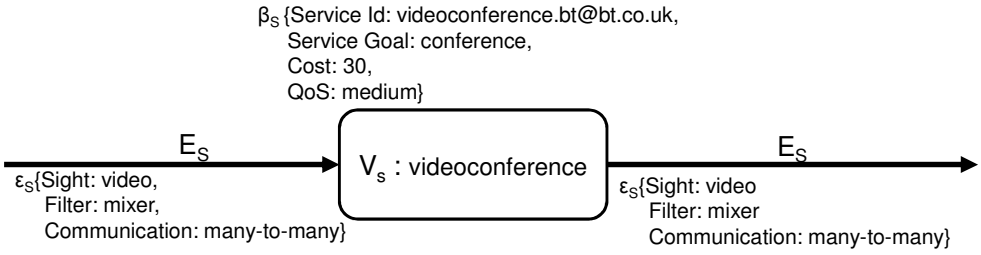


Figure 5.24: Graph representation of a service

The algorithm then checks if the first service has a successor service (see Figure 5.23). If it has a successor, the output(s) of the service itself and the input(s) of the successor are tested if they match against each other. For example the *newsticker* is the successor of the *videoconference* and their inputs and outputs are matching (see Figure 5.24 and

Figure 5.21). The two services then will be stored as a composition. After that it is checked if the last service of the stored composition has a successor. If it has a successor, the same as described will be processed recursively else it is checked whether a following group exists. If next group exists the above described procedure will be starting with the first service of this group.

It must be mentioned that sometimes there will be no matching services inside of the service repository, for instance a service that will not match with its output to the input of the a following service because of different media types, as well as no service for the given keyword exists. If this happens, the algorithm will also calculate so-called truncated service compositions, which comprise only of a part of the full service composition. The resulting compositions are sorted by completeness. This means that the complete ones are at the top of the list and that truncated compositions which will consist of one single service will be the last in list. Finally the algorithm connects the groups and sorts the resulting compositions again by completeness. The connection of the groups may depend on the reuse of a service if a unique identifier is specified, which means that the same instance of the service has to be used. By using the same instance of a service it can be ensured that no in- or output will be used twice. It is also checked that all in/outputs are linked if a service has multiple in- and outputs which are not optional. The following example will clarify this.

A given service composition request *(videoconference.name|newsticker) (name|recorder)* defines to reuse service *videoconference*, which means to use the same service instance that is the same conference room for a conference service. This implies that service *videoconference* must have multiple outputs. Two different cases have to be considered. In the first case, the outputs of service *videoconference* are

optional or only two of them are mandatory. In this case service *videoconference* can be linked against service *newsticker* and recorder. In the second case service *videoconference* has more than two mandatory outputs, which results in that the service composition could not be arranged, because one mandatory output is not linked.

The composition building parts of the algorithm are generally based on the depth-first search. The results given from the algorithm can be represented as a directed hypergraph *H={X, E}* where *X* is a set of services and *E* is a set of non-empty subsets of *X* (see Figure 5.25). The subsets of *X* include paths that are representing linked services, whereas the hyperpaths or hyperedges included by *H* will represent the service compositions.

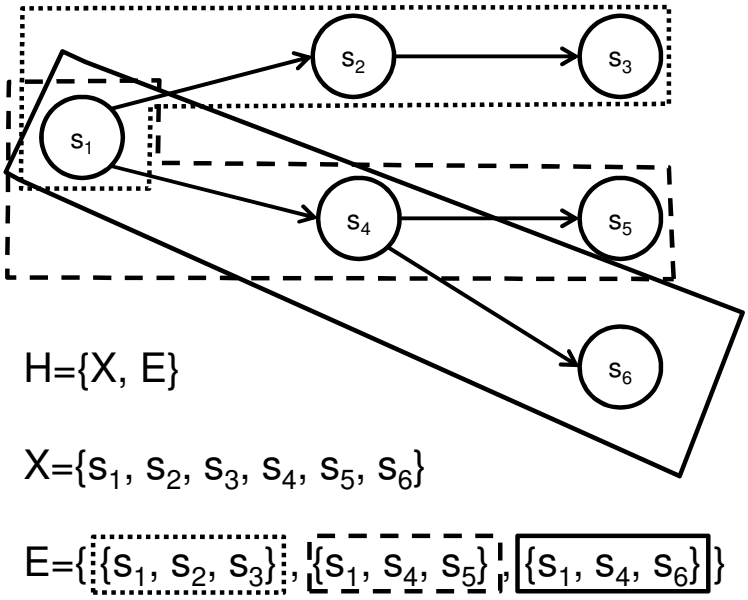


Figure 5.25: Hypergraph and hyperpaths

An example can be the set of services *X={im@bt.co.uk, translate.att@bt.co.uk, translate.orange@bt.co.uk, tts.en.@bt.co.uk, tts.de@bt.co.uk, tts.it@bt.co.uk}*. Here

the service *im@bt.co.uk* is an instant messaging service. The two translation services *translate.att@bt.co.uk* and *translate.orange@bt.co.uk* might have different language support, and the three text-to-speech services *tts.en@bt.co.uk*, *tts.en@bt.co.uk* and *tts.en@bt.co.uk* also will only support special languages. The resulting subsets of *X* could be *E={{ im@bt.co.uk, translate.att@bt.co.uk, tts.en.@bt.co.uk }, { im@bt.co.uk, translate.orange@bt.co.uk, tts.de@bt.co.uk }, { im@bt.co.uk, translate.orange@bt.co.uk, tts.it@bt.co.uk }}*.

After calculation of the resulting service compositions, the algorithm might sort them again regarding to eventually specified *non-functional properties*, e.g. regarding to QoS parameters. This means that for instance the QoS of all the services within one composition will be summarised. This is done for all the resulting compositions. After that the sums are compared and the composition with the lowest sum will be the one with the best QoS. Instead of the QoS also costs or other properties could be used. The algorithm finishes by returning a response to the requester, which contains a list of sorted service compositions represented by SIP URI chains, from which a SIP Route header can be compiled.

5.5 Conclusion

Different service description languages have been presented within this chapter to compare them against each other and the proposed SDL. This novel service description language has been introduced, which supports the necessary functionalities to describe a service, so that humans as well as machines can grasp what these services will do. Also the service description language provides the possibility to build service compositions on basis of connectors and their media types, and service compositions

can be specified with the service description language. As a result the novel SDL, was identified to fulfil all characteristics (see section 5.3).

Section 5.4 introduced an algorithm that has the ability to compile service compositions using keyword defined service requests as input. This algorithm also supports a ranking by different non-functional properties of the resulting service compositions.

A solid basis has been defined to implement a working service composition framework, but further essential questions with respect to service design recommendations arise. For example, how can services recognise whether they are part of a composition or not and how do the services identify what to do with the media streams, for example modify data in SDP? These aspects will be discussed in the next chapter.

6 Service design recommendations and restrictions

This chapter discusses novel scientific findings regarding to service design recommendations and restrictions in service composition. Some further requirements have to be fulfilled by the framework in terms of incorrect service behaviour caused by service interaction. In section 6.1 service design criteria are discussed based on different service characteristics that are identified during this research work, also limitations regarding the combination of services are defined. In section 6.2 the feature interaction and its resulting restrictions are outlined based on results of research publications and standardisations for SIP-based services.

6.1 Service design recommendations

This section will define different recommendations for service creation within the research's scope for service composition. As described in section 3.2, there are four classes of services: bearer, teleservices, supplementary and value-added. The bearer service here is IP and forms the basis for all other services; so all services, either teleservices or value-added services, with or without supplementary services, are built on top of IP. Supplementary services are mainly based on and implemented already with SIP, e.g. call forwarding on busy (CFB) or call blocking (CB). This means they belong mostly to the signalling provided by SIP. In addition to the mentioned service types, the following characteristics are identified with respect to the proposed service composition framework, which are based on scientific findings according to (Seidel, 2011).

- non-combinable services: these are services that can only run in stand-alone mode. This type of services cannot interact with others. In general these are services that are not designed to be combinable, for instance a service provider is not willing to provide service composition.
- non-stand-alone services: these are services which can only be used in combination with others (e.g. Calling Line Identification Presentation (CLIP) as a supplementary service or a service which overlays a subtitle onto a video).
- services with multiple end points: a service which normally is used in communication scenarios with more than two end points (e.g. conferences or broadcast services).
- services depending on special request methods: a service which will be requested by calling a different method than INVITE (e.g. MESSAGE or INFO).
- media processing services: a service that modifies, terminates or forwards media (e.g. transcoding service).

The above-named service characteristics will be discussed and recommendations will be presented for a better support to combine services in sections 6.1.1 - 6.1.4.

6.1.1 Non-stand-alone services

All non-stand-alone services should have a precondition that indicates the service might not be used in stand-alone mode and all further requirements referring to services that might be connected. For instance, it is impossible to use a service such as a supplementary service like CLIP as a stand-alone service. In most cases it is possible to extend a non-stand-alone service that it can be also used in stand-alone mode.

Therefore the non-stand-alone should be implemented with two different service capabilities to realise the utilisation of the service as stand-alone service; these capabilities are depending on an initial test. This test has to identify if the service is involved in combination with other services or not. If the service detects that it is in combination with another service fitting to its preconditions, then the service acts in normal behaviour. But if the service identifies that it is requested without a linkage to a suitable service, an additional behaviour has to be implemented for the service to support the functionality of the service even if a fitting service in combination is missing. An example for this purpose could be a service which overlays a subtitle on a video, e.g. for karaoke. If the subtitle service detects no video service for combination, it should have implemented a special functionality (see Figure 6.1), for instance the creation of a background picture as a replacement for the video service, which is absent. In consequence the corresponding *connectors* have to be set as optional, in order for the algorithm (see section 5.4) not to drop this service.

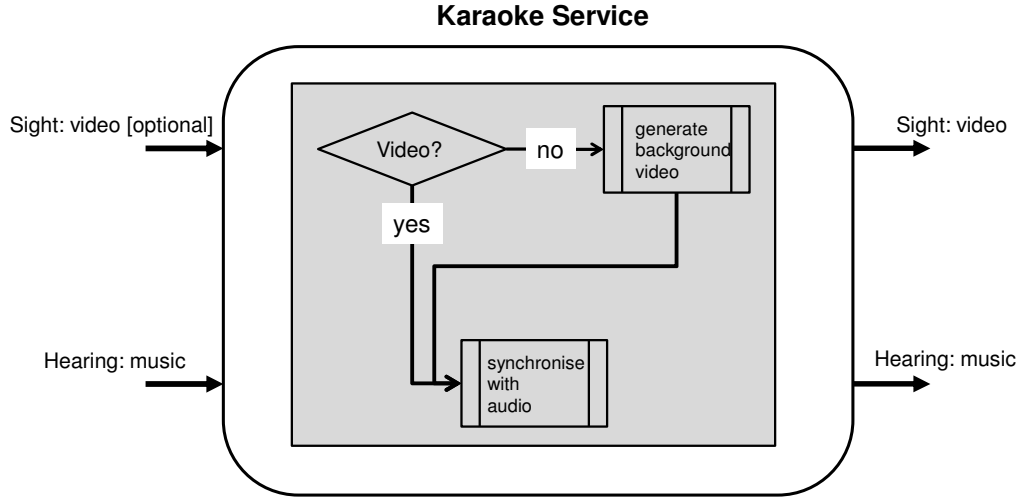


Figure 6.1: Adapted non-stand-alone service

6.1.2 Services with multiple end points

All services with multiple end points need to be fed with data describing the end points in detail (e.g. IP addresses and ports), because they will typically send or receive data to or from more than one end point. For that reason two different possibilities exist.

One recommendation of service design to provide a better support for service composition is the definition of preconditions, whereby the declaration of a set of end points will be ensured. This solution has one disadvantage, because another service has to be involved to collect the end point's data. The difficulty behind this is that during the composition process the appropriate service has to be identified automatically, which leads to a more complicated algorithm and the execution speed of the composition process will be reduced. So the second possibility has to be favoured, which is that the service logic is handling the collection of the needed data. The following will represent an example for illustration. Within that example a conference service needs further information about the participants that should be invited. As shown in Figure 6.2, the service sends a request to the initiator (here: a SIP MESSAGE). Within this request a description could be given to explain how the data can be supplied to the service. For instance a user could be asked to enter a web page and enter the end points SIP URIs.

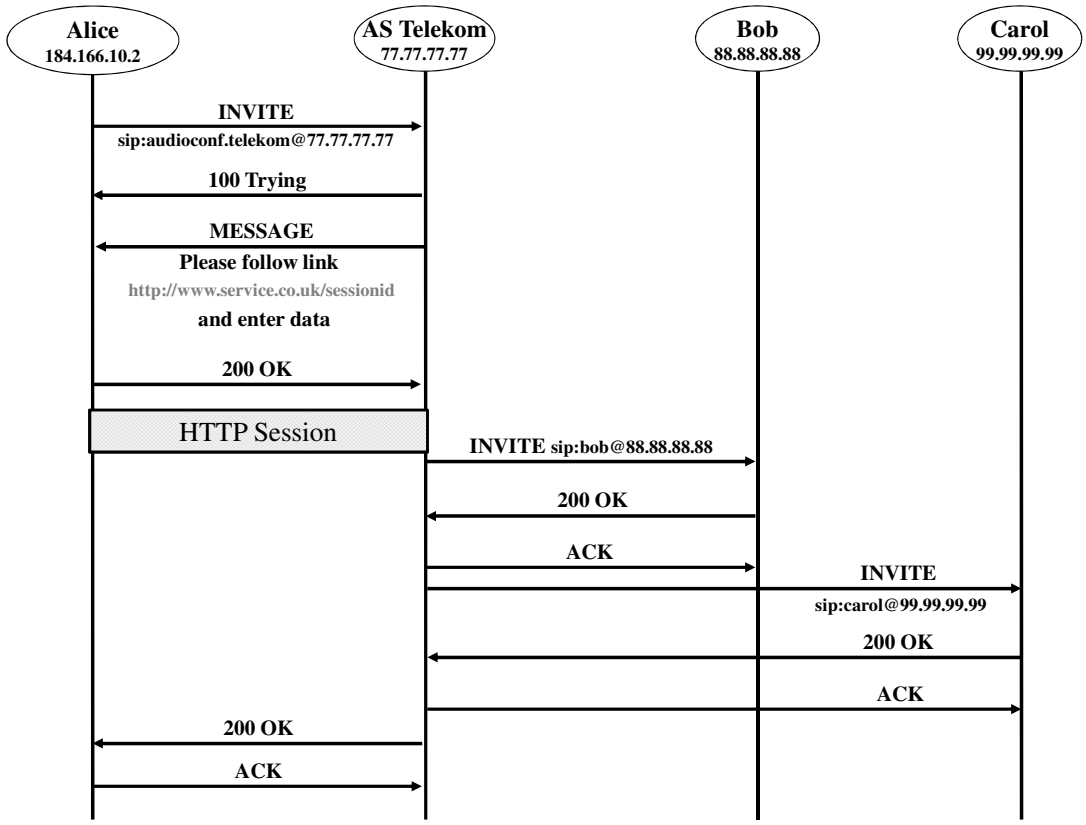


Figure 6.2: Exemplary service with multiple endpoints and data input through web site

As a result it can be said, that in generally, whenever a service needs further data input for its execution, the service behaviour should be extended, so that the service will be fed with the necessary data. The way this will be implemented depends on the choice of the service provider. It has to be emphasised that this recommendation is not only limited to services with multiple end points, it is also valid for any service which needs further data input.

6.1.3 Services depending on special request method

Additionally there might be services which are depending on the type of methods to be used as a request (e.g. SIP MESSAGE). An example is a text-to-speech service (TTS), which transforms text into synthesised speech. A defined precondition for this

service could be that the only acceptable request will be a SIP MESSAGE. This might result in incompatibility with other services, which have to be invoked by a SIP INVITE according to the proposed framework (see section 4.5.3). In this special case, the algorithm also has to be extended to check pre- and postconditions. Another possibility to provide a better support for compatibility the services should implement different service logics (see Figure 6.3).

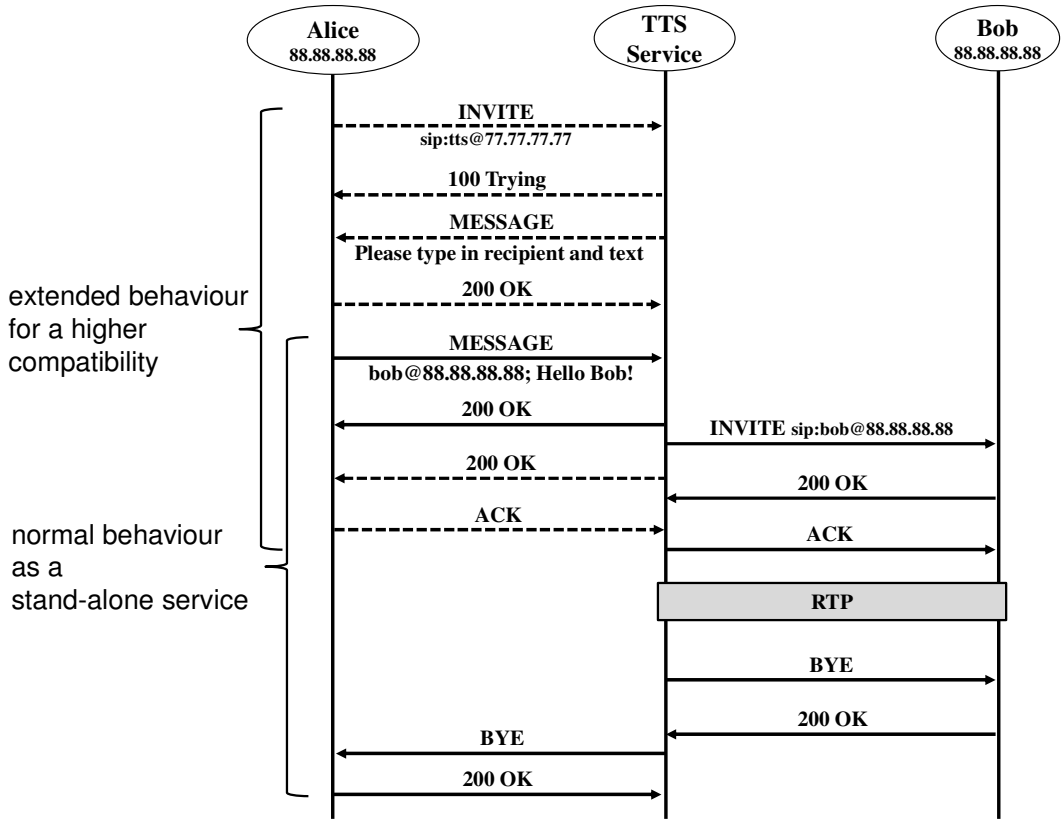


Figure 6.3: Adapted service which depends on special request method

The service logics could be the support of more than one SIP method (e.g. INVITE and MESSAGE). This might also be useful so that such a service will be enabled to be utilised as a stand-alone service.

6.1.4 Media processing services

A service that processes media, whether modification, termination or forwarding, has to analyse the media description part of a session, particularly the SDP part within a SIP request. Depending on the functionality of the service, some parts within the media description might be changed. For example, an IP address and/or port might be changed so that the service can modify the media stream (e.g. change size of video or add a subtitle). Concerning this matter, it is important to have knowledge if the service is utilised in combination with others or not. If such a service will forward the media stream to a terminating site (compare with Figure 5.22) it might be unusable as a stand-alone service, thus it might be conceivable that the service has the ability to identify if it is pegged into a service composition and can change its behaviour, so that the media stream will be forwarded to the originating site if it is not bound to a service combination. The arising question is, how a service can identify whether it is bound to a combination or not.

All services, not only media processing ones, should implement a (preferably simple) mechanism to detect whether the service is part of a service combination or not. As mentioned in section 4.5.3, the SIP Route header has to be used for service composition; consequently a service should check during its invocation whether it is part of a service composition request by parsing the SIP headers for the appearance of a Route header. The usage of the Route header is restricted to be used only within service composition requests applying the proposed framework. If a Route header is present within a SIP request, the service can be sure that this is a service composition request, only the last service of a service composition chain will not be enabled to identify whether it is a service composition request by parsing for a Route header,

because of the Route header removal by the penultimate service within the chain. A possibility to inform also the last service from a composition chain might be to set an identifier within a SIP header. Consequently, the second last service within a composition should add a parameter to the SIP Via header. As defined in (IETF RFC 3261, 2002): “A Via header field indicates the transport used for the transaction and identifies the location where the response is to be sent.” In other words the Via header is used to record the SIP route taken by a request then afterwards to route the SIP response back to the requests originator. Every SIP element visited by the request in between adds on top of the list of Via headers its own address and Branch parameter. Beyond the Branch parameter, further parameters can be added to the Via header. Thus the second last service within the service composition chain should add an identifier parameter, so that the last service of a composition chain also is enabled to identify that it will be used within a composition. An example SIP Via header with the via-extension “composite-id” is presented below:

Via: SIP/2.0/UDP audioconference.co.uk; branch=z9hG4bK74b76; composite-id=”1234”

The Via “composite-id” parameter is a “via-extension”, as defined by the SIP ABNF (Augmented Backus–Naur Form); see Section 25.1 of RFC 3261. The following is its ABNF regarding to (IETF RFC 4234, 2005).

via-sip-composite-id = "composite-id" EQUAL
*LDQUOT *(qdtext / quoted-pair) RDQUOT*

The presented “via-extension” parameter should be registered by the operators of the proposed framework at IANA (Internet Assigned Numbers Authority) to become accepted.

The recommendations which can be given for the design of services regarding to the proposed framework are summarised in Table 6.1. The table is organised in three columns; the first column characterises the given recommendation, the second column determines the impact or consequence of the recommendation, and the last column defines the characteristics of services that will fit on the presented recommendation. In general, if a service fits into more than one characteristic, it should follow all recommendations appropriate to the characteristics. For example, a non-stand-alone service should follow the recommendations utilisation of preconditions, implement different service capabilities and implement composition detection.

Table 6.1: Recommendations for services by different characteristics

Recommendation	Impact/Consequence	Service type/characteristic
precondition	algorithm has to consider precondition for returning the results for a service query	non-stand-alone service
different service capabilities (adaptive service)	corresponding connector should be set as optional along with preconditions	non-stand-alone service
data collection through service logic	extended service behaviour depending on the implementation	service with multiple end points service which needs further data for its execution
multi request method support	extended service logic to handle multiple request methods	service depending on special request method
composition detection	extended service logic to detect if it is in composition or not	media processing service non-stand-alone-service (should be generally used by any service)

6.2 Service interaction

In telecommunication networks, feature interaction occurs when several features or services, operating simultaneously, interact in such a way as to interfere with desired operation of some of the features or services (Lennox and Schulzrinne, 2000). Service

interactions are especially common in telecommunications, because all services are modifying or enhancing the same basic service (e.g. bearer or teleservice) (Zave, 2004). Also in IP-based telecommunication systems service interaction is a well-known problem (Lennox and Schulzrinne, 2000). The impact of this problem will increase because of the growing number of services, especially in connection to the proposed network within this research work. Services from multiple service providers, deployed on peers, will likely result in service interaction, so this problem has to be discussed here. Also, end-users may be enabled to specify their own services on their terminal equipment. The terminal equipment may be equipped with the possibility to define and activate different supplementary services, e.g. call forwarding when busy, call blocking by utilisation of a black list, voicemail or automatic redial. Thus end-users are permitted to personalise the call control, which might result in unwanted service interaction (Kolberg and Magill, 2001). A number of publications discussing service interaction have been published as part of the Feature Interaction Workshops since 1994, see (Bouma and Velthuijsen, 1994), (Cheng and Ohta, 1995), (Dini *et al.*, 1997), (Kimbler and Bouma, 1998), (Calder and Magill, 2000), (Amyot and Logrippo, 2003), (Reiff-Marganiec and Ryan, 2005), (Bousquet and Richier, 2007) and (Nakamura and Reiff-Marganiec, 2009). Various solutions for feature and service interactions were already discussed at the Feature Interaction Workshops. Regarding to this research work possible solutions based on the outcome of the Feature Interaction Workshops, especially (Calder *et al.*, 2003), will be presented in the following.

In (Calder *et al.*, 2003) different solutions, software engineering approaches, formal methods and on-line techniques, for service interactions were discussed. The software

engineering approaches address the elimination of interaction during the service creation and development process, which already was stated in section 6.1. Formal methods primarily serve as interaction detection and they are invariably off-line, which means that they are utilised not during the run-time. The approaches of formal methods were not examined, because the focus is not on detecting feature interactions it is on resolving them. On-line techniques were also presented in (Calder *et al.*, 2003), which are suitable to the proposed framework. These on-line techniques provide usually a combination of detection and resolution mechanisms. A detection of an interaction is only useful if the problem can be resolved at runtime (during execution), which should be possible within the proposed framework. Consequently, mechanisms for interaction detection and resolution are illustrated.

Before discussing these mechanisms, the two most important groups of on-line techniques are presented according to (Calder *et al.*, 2003). The mentioned groups are divided into location of control and type of information.

Location of control:

- Feature Manager based approaches. These are based on mostly centralised elements called Feature Managers, which observe and control features. One disadvantage applying this approach is that the Feature Manager has to be involved into the call control. This is against the decentralised approach of this research work.
- Negotiation based approaches. Here the features (or services) are enabled to communicate to each other to detect and potentially solve the interaction

problems. If necessary the conflict can be forwarded to a special service which will resolve the conflict.

Type of information:

- A-priori information. This approach rests upon data collected during the design-time. For this purpose empirical values will be used, which can be represented as a matrix relating all services. The amount of information grows linearly per service.
- Isolated on-line environment. Here the information is gathered at runtime in a separate closed environment. This is not practical for live detection and resolution.
- During run-time. The information is collected during the run-time of the feature.

As result the qualified on-line techniques are the negotiation based approaches, a-priori information and during run-time information. In the following these three techniques will be presented.

Negotiation based approach

Beside the possibility to negotiate interactions based on the Offer/Answer model with SDP (IETF RFC 3264, 2002) further SIP extensions were specified which might also act as negotiation mechanisms for service interaction. First of all, the Option Tags have to be mentioned. These will be used in header fields such as Require, Supported, and Unsupported (IETF RFC 3261, 2002). The Option Tags are specified in additional RFCs which can be looked up from the list at (Roach *et al.*, 2013). With these tags, the User Agent Client (UAC), the one who sends the SIP Requests, should include a

Supported header if it supports extensions to SIP, so that the User Agent Server (UAS), the one who receives the request, will be informed about the supported extension. If such an extension is demanded by the UAC it must insert a Require header. By contrast the Unsupported header field should be used by the UAS to list the features which are not supported. The defined Option Tags transported in the mentioned header fields specify a variety of features. One powerful Option Tag is the *pref* Tag, which is used in combination with the Require header to ensure that the caller's (UAC) preferred extensions are supported. Regarding to the *pref* Tag and the referenced standard (IETF RFC 3840, 2004) the services are enabled to convey their capabilities and characteristics to other services and User Agents. The information is conveyed as parameters of the Contact header field. For instance here a service can note which media it supports. The following example, see Figure 6.4, clarifies this.

To request a User Agent or service for supported characteristics and features the SIP method OPTIONS should be employed. Assumed that the videoconference service and newsticker service as already stated in section 5.2 should work in combination, the newsticker service requests the videoconference service for its supported characteristics before forwarding the SIP INVITE (see section 4.5.3). Within this OPTIONS request the newsticker service sets the Require header field with the *pref* Tag. Subsequently the videoconference service sends a reply for that request, which contains the supported characteristics, e.g. video supported and the following SIP methods are supported (INVITE, BYE, OPTIONS, ACK and CANCEL). Thus the newsticker service can identify if all necessary characteristics are supported by the videoconference service. If one needed characteristic is missing then the call request

should be cancelled. On the basis of this little extension, service misbehaviours can be eliminated.

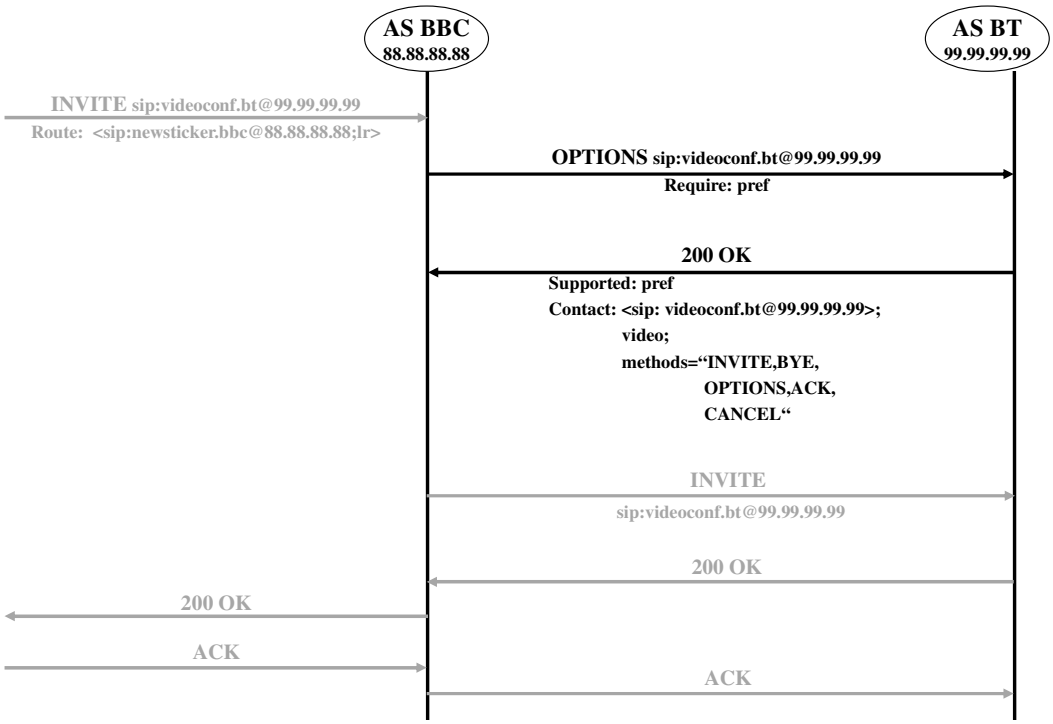


Figure 6.4: SIP-based service capability negotiation

Additional SIP header fields, such as Allow, Accept, Accept-Language and Accept-Encoding, already convey some information about the capabilities of a User Agent or service. These header fields also can be used during service invocation to identify if special capabilities such as audio, video or mobility are supported. In dependency of the allowed or accepted capabilities a service combination might be established. This also is based on the usage of the SIP OPTIONS method as described above.

A-priori information

Here the service interaction could be managed through the service description language *conditions*. The service developers are allowed to add information with respect to other services. For example, they can define that designated services are cannot to be directly connected with this service or that it is forbidden to utilise this service in combination with other services of a special kind. An example might be the combination with a supplementary service, e.g. the supplementary service call forwarding when busy and a voicemail service (Kolberg and Magill, 2001). The interaction between the two services arises when a call arrives while the user is busy. For example if the user is invited to a conference and he is busy, it will not help to forward to the voicemail service. The question which comes up is, is the voicemail service compatible with CFB or not, means should the call be forwarded to and accepted by the voicemail service. This can be defined within the *conditions* (pre- and/or postcondition) of the SDL (see section 5.2). So it can be specified as a postcondition of the CFB, whether a voicemail service is supported or not, or it can be specified as a precondition of the voicemail service whether it is compatible with CFB or not in conjunction a conference service. Therefore the algorithm as specified in the former chapter (see section 5.4), should be extended to consider conditions during the creation of service compositions.

During run-time information

The collection of information during the run-time is necessary. This can be handled by utilising the presented negotiation based approach. Another possibility is the collection of information during the service composition process, if the service composition

server is extended with a new functionality. The composition server has to request further service characteristics before combining the services. For this purpose every service that might be part of the resulting composition will be requested with the SIP method OPTIONS and the *pref* Tag. Depending on the replied characteristics the service will be part of the composition or not.

All of the presented possibilities will help to reduce service interactions, but the a-priori information based approach should be preferred. The negotiation-based approach has one disadvantage: the chosen service composition might not be runnable, if cancelled by a service in between. Also the information during gathering run-time has one disadvantage. The process of creating the service compositions will be slowed down, because previously every possible service has to be requested for its characteristics. As a result, the a-priori approach should be implemented, because all necessary information can be stored within the service description of each service. Figure 6.5 illustrates a formal notation of the presented on-line techniques, which also can be utilised in combination to minimise possible service interactions.

Figure 6.5 helps to find an optimised way to reduce service interactions. Following the steps from the presented formal notation, the first question is if information is already available before composing the services, then the a-priori information will be used to form the composition additionally information may be acquired during the composition process by the composition server, therefore each service has to be requested by a SIP OPTIONS for its characteristics. If not the User Agent which will invoke the services for the composition may additionally utilise SIP OPTIONS to request the characteristics of each service to identify if an interaction may occur.

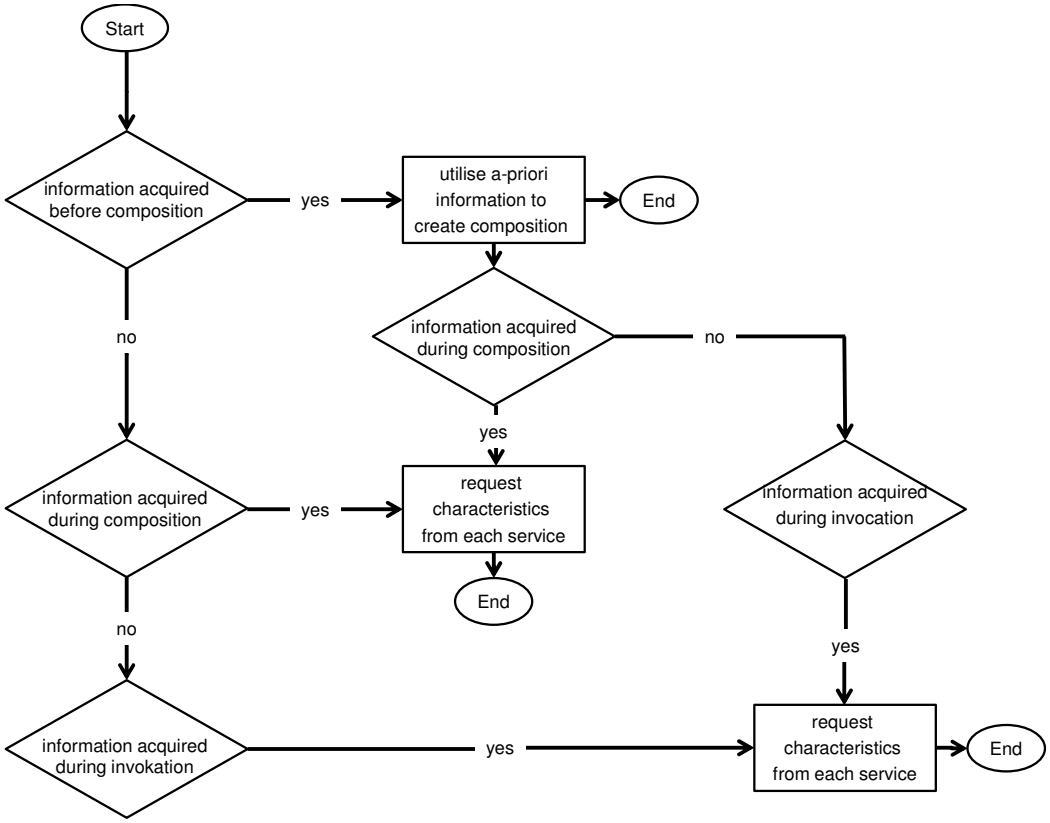


Figure 6.5: Formal notation of presented on-line techniques to reduce service interactions

If no information about the services was available at the beginning, the composition server can request the characteristics from each service prior to create the composition. Finally if none of the above described possibilities had been utilised then the User Agents should request the characteristics of the services by its own to avoid unwanted interactions.

6.3 Conclusion

Within this chapter, further recommendations regarding to service design and interaction have been presented. The framework can be extended to eliminate many lacks with the illustrated approaches. Also general recommendations have been given to identify composition dependencies during service invocation. Furthermore advices

have been defined to enable services to recognise what to do with the media data, e.g. forwarding or modifying. Also general problems concerning service interaction have been highlighted. Finally, a procedure has been presented that is the result of research publications and standardisations for SIP-based services. Following the three stages of the procedure service interactions can be reduced or even solved.

7 Research prototype and framework evaluation

This chapter describes the prototype implemented as a proof-of-concept evaluation to demonstrate the feasibility and efficiency of the proposed framework. The prototype consists of a set of different software components that provide all necessary elements to illustrate the core subset of modules that constitute the framework: service provisioning, discovery, binding, publishing and composition, as introduced in chapters 4-6. Section 7.1 summarises the criteria to fulfil for the overall concept then, section 7.2 depicts the architecture and utilised software components and their functionality. Section 7.3 discusses the utilised functional tests for the validation of the proposed framework regarding its applicability for service composition and section 7.4 concludes the chapter with a critical discussion on the overall evaluation of the framework.

7.1 Criteria for evaluation

The following presents all criteria that have to be fulfilled by the framework, all discussed in the previous sections. The different criteria are classified under the following five main features: general features, NGN key features, SOA features, Service Delivery Platform characteristics and Service Description Language characteristics.

1. General features

The core elements of the framework have to utilise a peer-to-peer infrastructure. This implies that no central database exists; so every peer stores and provides a part of the available data of the overall system, but none of these peers manages or is aware of the complete data pool. No central coordination exists this means that there is no central node which controls the interaction of the peers among themselves. Every peer has client and server functionalities and is autonomous in its decisions and manners. The whole stored data of the system has to be available, in spite of distributed storage and/or possible connection failures. So the data has to be available at any time by any peer. Furthermore SIP has to be utilised by the peers for service invocation. Existing services of a SIP based communication network, basic services and supplementary services, have to be supported. The whole service concept of the framework has to be based upon the layered concept of services in telecommunication. And last but not least the support of arbitrary media, e.g. audio, video and/or text, has to be fulfilled.

2. NGN key features

The service related key features of the NGN concept have to be fulfilled. These key features are applicability for arbitrary services, separation of call/service control and media data transport, application server support and support for multimedia services. Furthermore data transport has to be packet-based; the whole framework has to be scalable and unrestricted access for users to different service providers has to be enabled.

3. SOA features

The following features inherited from SOA regarding to services have to be maintained. Services have to be loosely coupled, location independent, reusable, modular and composable. Furthermore the discoverability of services has to be given and the services have to be described and accessible by implementation independent interfaces. Finally the “find, bind and execute” paradigm has to be applied.

4. Service Delivery Platform characteristics

Some of the essential characteristics of a service delivery platform have to be provided. The support of services provisioning has to be given also the possibility to combine or create services out of existing services has to be supported. The interfaces of the services have to be specified so that the services are accessible by the interface descriptions. Furthermore the complexity of services should be hidden by abstraction of interfaces and finally the SOA concepts have to be supported.

5. Service Description Language characteristics

Also a number of characteristics that have to be fulfilled by the utilised SDL which forms a service interface have to be fulfilled. This SDL has to be interpretable by machine and it has to contain a human readable description so that a human is able to understand the functionality of the service by this description. The entire SDL has to be platform and programming language independent and has to be based upon standardised languages/technologies, e.g. XML. In addition the SDL must have the possibility to specify the functional and non-functional properties of a service. Aside from that the service’s functionality has to be characterised by utilisation of simple

keywords. Lastly the SDL must have the ability to support any human perceptual service, for instance a service which conveys smell or taste.

All of the named criteria will be utilised to evaluate the framework and its functionalities.

7.2 Framework prototype architecture and implementation

The research prototype has been developed to demonstrate the essential framework's functionalities as service provisioning, discovery, binding, publishing and composition. To be as near as possible to a real-world implementation, the whole framework has been set up on virtual machines based on Oracle VirtualBox [Oracle 2013] connected to each other through the graphical network simulator GNS3 [GNS3 2013]. The GNS3 enables the simulation of networks by emulating Cisco hardware for routers and possibly switches. It is also possible to connect real networks and virtual machines to the simulated network. Figure 7.1 gives an overview of the prototype architecture that is based on GNS3 and virtual machines. As illustrated in Figure 7.1, four IPv4 class C networks are connected to each other by a router. In the network 12.0.0.0/24, a SIP User Agent is connected which is hosted on the underlying operating system. The SIP User Agent will be utilised to request service compositions and invoke later on one of the returned compositions as described in sections 4.5.2 and 4.5.3. The services themselves are provided by SIP application servers (see section 4.5.1) belonging to the network 14.0.0.0/24. A dynamic DNS server and the DHT bootstrap peer are hosted on a server in network 18.0.0.0/24. The core elements, SIP Proxy,

Registrar, discovery and composition server, of the framework are supplied within network 16.0.0.0/24.

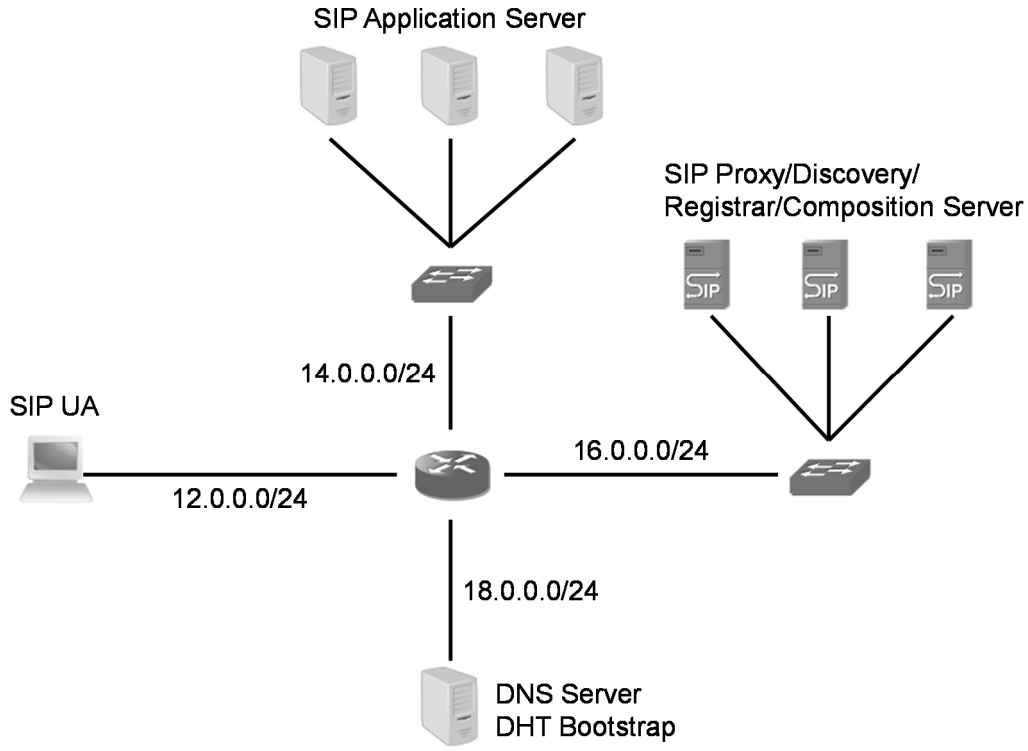


Figure 7.1: Prototype architecture

All servers are implemented by virtual machines utilising Oracle VirtualBox (see Figure 7.2). The SIP User Agent makes use of the VirtualBox host-only Ethernet adapter, which acts like a "loopback" interface on the host. VirtualBox creates a new software interface on the host next to the existing network interfaces. Through the newly created host-only adapter the virtual machines that are connected to cannot talk to other networks connected to the host. GNS3 links the SIP User Agent and the virtual machines together by a router on IP basis. Within Oracle VirtualBox three different internal networks were defined to separate the servers from each other. Based on an internal network a virtual machine can directly communicate to other virtual machines

on the same host connected to the same internal network; so all virtual machines of one internal network are connected to each other by a virtual switch.

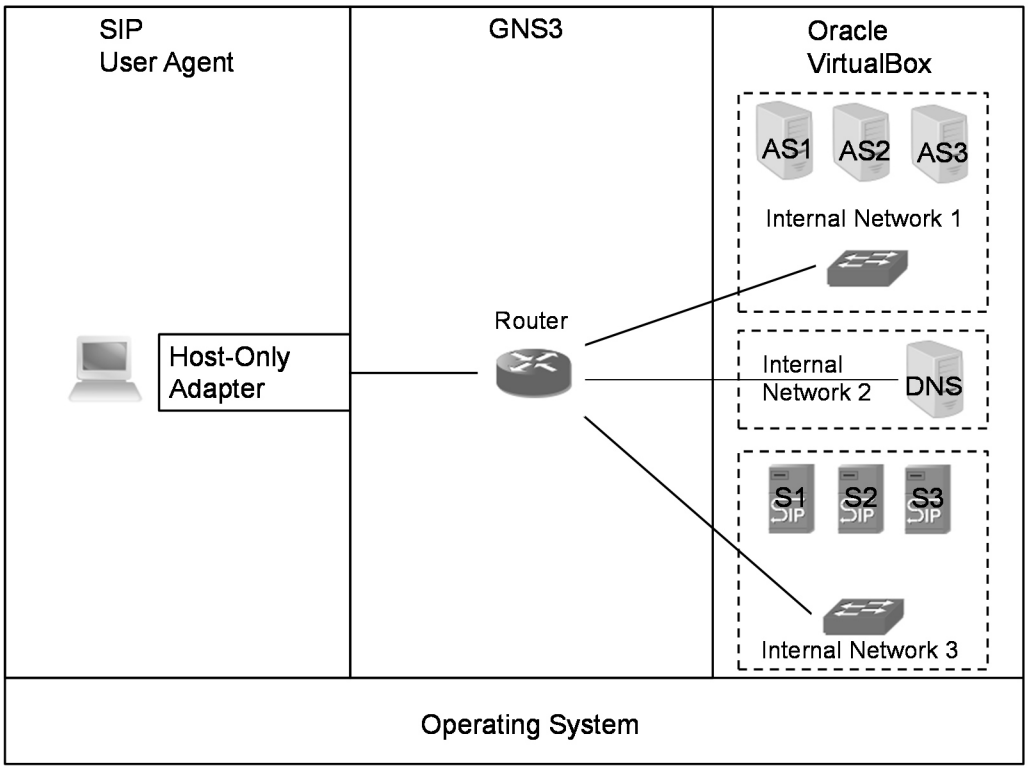


Figure 7.2: Prototype software architecture

All virtual machines are based on Tiny Core Linux [Shingledecker 2008] as operating system. Tiny Core Linux is a very small Linux distribution, which means that it needs only a small amount of megabytes to run, such as 128 megabytes, which is why it was chosen for the virtual machines. BIND (Berkeley Internet Name Domain) [ISC 2013] was utilised to form the dynamic DNS server.

All other software used for implementation made use of the programming language Java and further libraries. The libraries utilised are OpenChord, JAIN SIP (Java APIs for Integrated Networks), DNSJava, JAXB (Java Architecture for XML Binding), Gstreamer-java and OpenJSIP. The OpenChord API (Distributed and Mobile Systems

Group of Bamberg University, 2008) is an implementation of the Chord (Stoica *et al.*, 2001) distributed hash table and was utilised to implement the peer-to-peer functionalities within the discovery, Proxy and Registrar servers (see Figure 4.4). OpenJSIP (OpenJSIP, 2013) is a Java based implementation of SIP Proxy, SIP Registrar and Location server which is based on the JAIN SIP API. The Location server was replaced by an OpenChord implementation, so that the location service is used as described in section 4.1. Furthermore the main functionality of the composition server has been implemented based on JAXB to realise the algorithm applying XML as service description language's base. DNSJava was applied by the SIP Proxy, Registrar, discovery and composition server to automatically update the entries within the dynamic DNS server. The servers will append them self to the DNS so that they are reachable, if one of them is going offline caused by an error or wanted, the other servers will detect that and will instantly remove the entry for the offline server from the DNS.

The application servers are implemented on basis of JAIN SIP for the SIP communication and they utilise Gstreamer-java for the media handling and RTP transmission. Also the SIP User Agent is based on JAIN SIP and Gstreamer-Java. The following Figure 7.3 illustrates the architecture of the SIP application server. The SIP application server utilises the SIP and SDP stack of JAIN SIP and implements on basis of this the SIP UAC (User Agent Client) and SIP UAS (User Agent Server) functionality. On top of this a service dispatcher has been implemented, which handles the delivery of the incoming SIP requests and forwards them to the requested service (e.g. by its SIP URI).

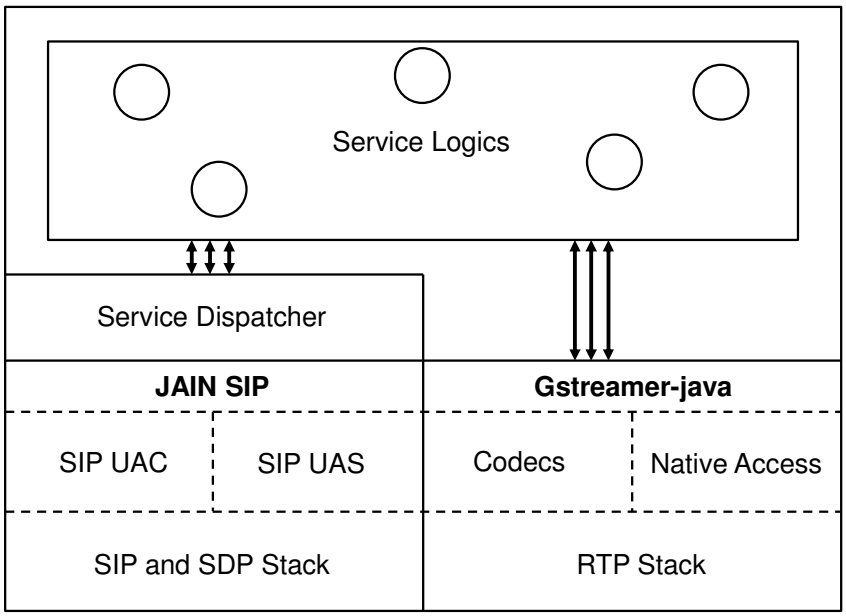


Figure 7.3: Application server architecture

Within the service logic the services were realised. These might have control over Gstreamer elements such as codecs, or RTP specific packetizers, which will pack the chosen codec into RTP packets. Gstreamer-java therefore needs to have access to a so-called native library, which is a library written in the programming language C. The native library is the Gstreamer API (Gstreamer, 2013), which is a well-known open source multimedia framework.

The implementation consists of five Java projects: Bootstrap, SIP servers, SIP application server, SIP User Agent and dynamic DNS manager. In all projects the source code that was created or modified are arranged within Java packages named "de.fhffm.research.*". All service logics and SIP behaviours in the SIP application servers and SIP User Agent are self-developed. In Table 7.1 the overall Java code statistics are shown, which were calculated by the Eclipse plug-in Metrics (Metrics2, 2014).

Table 7.1: Java code statistics

Project name	Number of Java classes	Total lines of code
Bootstrap	2	54
SIP servers	126	6125
SIP application servers	51	5735
SIP User Agent	15	2387
Dynamic DNS manager	3	285
total	**197**	**14586**

As an example for a service implementation the newsticker service will be presented here. Before a service can be invoked a service query has to be created (see Figure 7.4). Here a service with the keyword *information* is requested. The service request message SUBSCRIBE is sent to the discovery server. The discovery server then answers with NOTIFY. Within the body of the NOTIFY request the identified services matching the keyword *information* are returned.

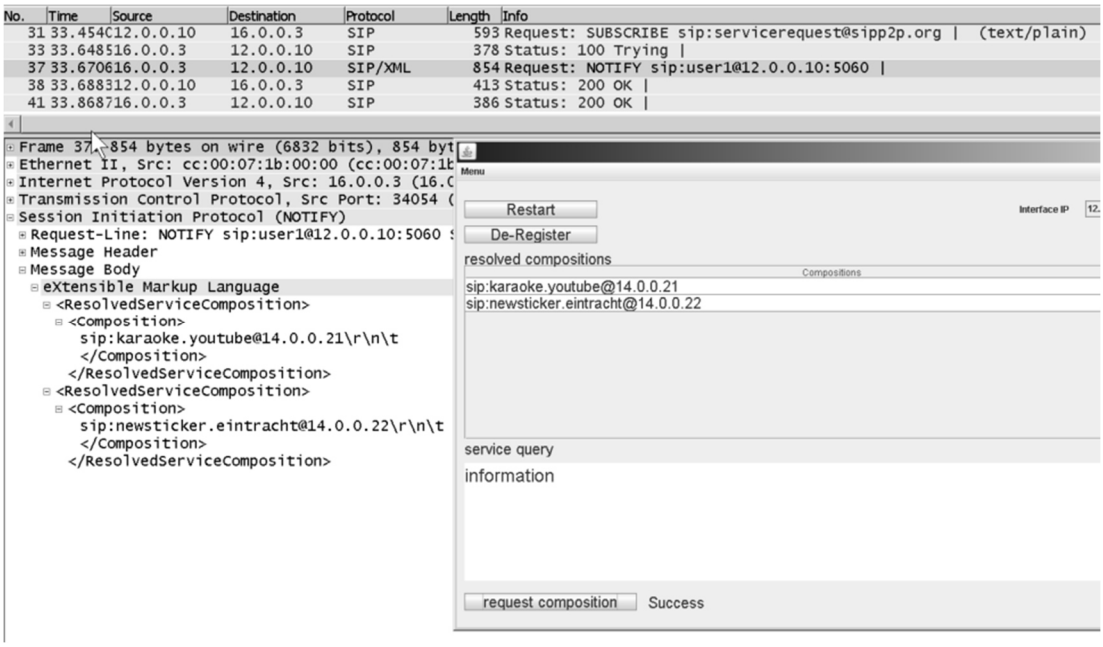


Figure 7.4: Service query implementation

Assuming that another service query was created, for instance *playback|information*, the result is illustrated in Figure 7.5. The second resolved service composition is chosen and invoked. As a result a music video with an embedded newsticker is called.

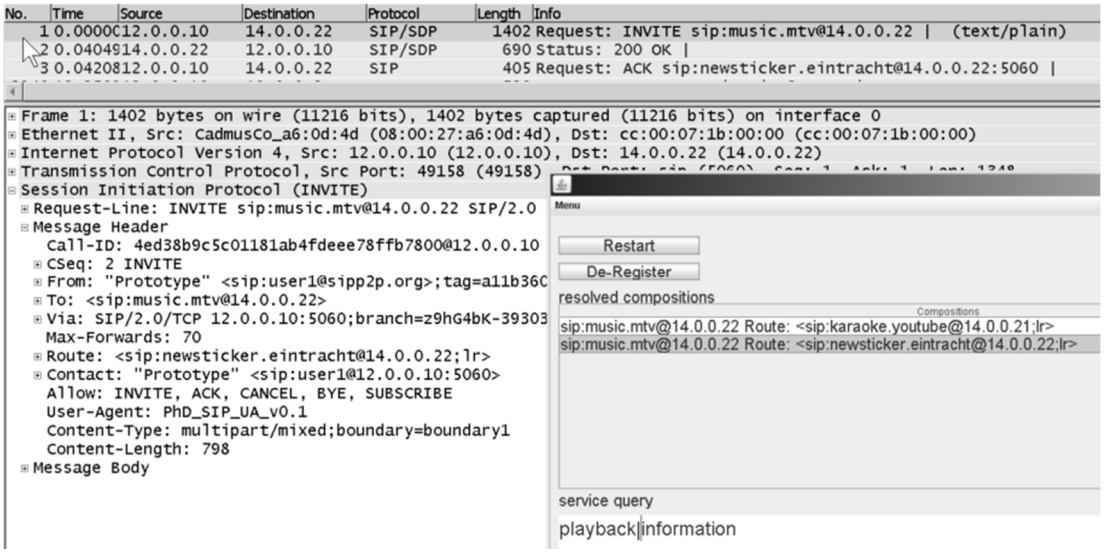


Figure 7.5: Service composition example

The resulting newsticker embedded to a music video is depicted in Figure 7.6.

Figure 7.6: Embedded newsticker service

In this service example the music video is transferred by the playback service to the newsticker services. Therefore the playback service encodes the video data and packetizes it into RTP transported over UDP. Next the newsticker receives the data

and depacketizes and decodes the video data. Then a substring of a hardcoded news text is overlaid the video data. Every 150 ms the substring will be actualised as represented in Figure 7.7. The video data will be encoded and packetized into RTP and sent over UDP to the client.

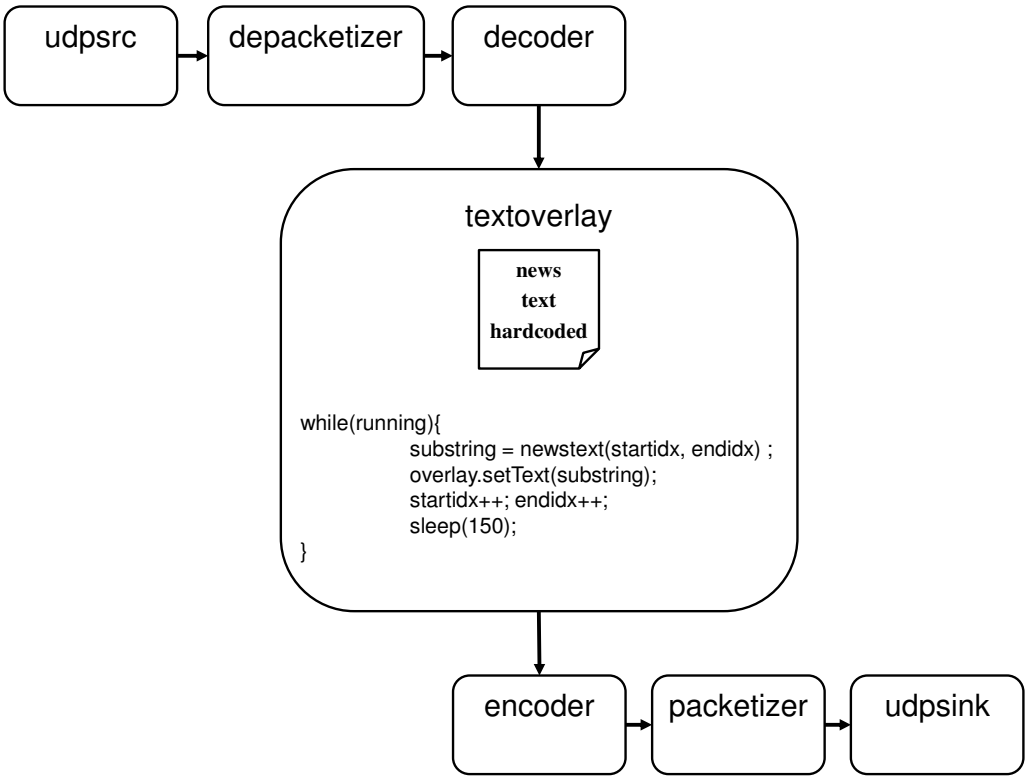


Figure 7.7: Schema of newsticker implementation

The function blocks presented in Figure 7.7 are named by the used Gstreamer elements. All the functionality, such as text overlay, is done by Gstreamer, which means that Gstreamer is responsible for how to overlay a text onto a video.

The whole framework's software collection and the virtual machines are included on the DVD-ROM enclosed with this thesis so that the research prototype can be run.

7.3 Validation of implemented prototype

To validate the framework blocks and functionality, a number of test scenarios were implemented into the prototype. Furthermore, the algorithm functionality that forms compositions has been validated separately on basis of predefined service descriptions and tests, without the implementation of existing services for the service descriptions. Section 7.3.1 presents the testing of the implemented algorithm. In section 7.3.2 the scenarios to test the functionality of the framework are illustrated.

7.3.1 Test scenarios for the algorithm implementation

The algorithm was tested on the basis of simulating an incoming service composition request. Therefore a set of XML-based service descriptions has been defined with the following keywords and in-/outputs (see Table 7.2).

Table 7.2: Testing set of keywords and in-/outputs

Keyword	Identifier	Inputs(number)	Outputs(number)
announcement	announce.tcom	none	speech(1)
conference	conf.polycom	video(6)	video(6)
conference	conf.tandberg	speech(6)	speech(6)
Control	selector.behringer	speech(4)	speech(1)
Gaming	spaceinvaders.atari	dataType:Sensor(1)	video(1)
Gaming	pingpong.atari	dataType:sensor(2)	video(2)
information	information.greatwestern	none	speech(1)
information	information2.greatwestern	none	animation(1)
Multiplier	multi.technics	speech(1)	speech(4)
Playback	play.fm	dataType:file(1)	video(1); music(1)
recognition	recogniser.att	speech(1)	text(1)
recognition	recogniser.bt	light(1)	text(1)
recognition	recogniser.telefonica	temperature(1)	text(1)
recording	record.google	speech(2)	dataType:file(1)
synchroniser	lipsync.adobe	speech(1); video(1)	speech(1); video(1)
text2speech	tts.1und1	text(1)	speech(1)
text2speech	tts.alice	text(2)	speech(1)
translation	translation-en-de.leo	text(1)	text(2)
Voting	vote.bt	none	dataType:information(1)
Voting	vote.eplus	none	dataType:information(1)
Voting	vote.o2	none	dataType:information(1)

As can be seen from Table 7.2, some of the illustrated services represented by their service descriptions are combinable. By utilisation of the already outlined syntax (see sections 5.2 and 5.4) multiple combinations of the listed services were compiled for testing purpose. The testing has been accomplished under the following points of view, which represent known problems during a service composition request.

- Unknown keywords are used within the request at any position of a composition, e.g. *(unknown|information|translation)*.
- The reutilisation of the same instance of a service has to be fulfilled by tagging a service with the syntax *.name* as a unique identifier, for example *(information.name=i1|translation) (i1|text2speech)*.
- The service composition query contains non-composable services, because of different inputs and outputs, for instance audio as output of one service should be connected to text as an input of the following service.
- Grouping of service chains and reutilisation of the same instance of a service has to be supported. An example is shown in Figure 7.8.

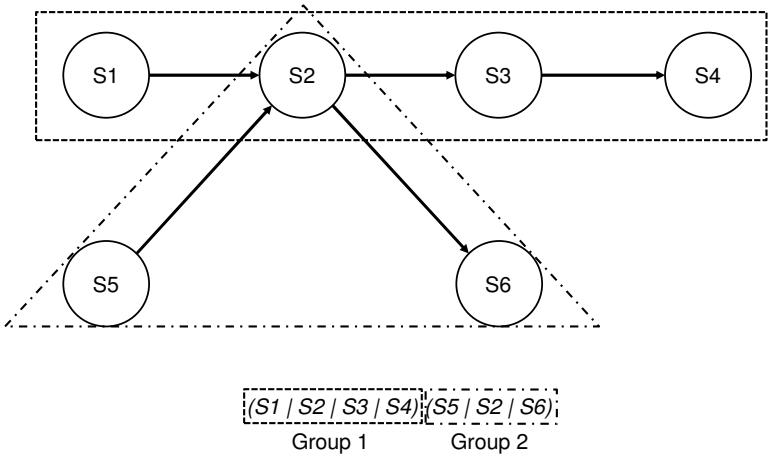


Figure 7.8: Grouping and reutilisation of services

- The possible resolutions for the service composition request have to be sorted and ranked by completeness. This means that all requested keywords could be resolved and a service composition could be built equal to the requested composition. Furthermore the resulting compositions have to be sorted from complete compositions down to partial compositions or in worst case down to single services.

The following exemplary request should illustrate a test for the implemented algorithm. Here only the request body is shown.

(null|information.name=i1|recognition|translation.name=t1)
(t1|text2speech.name=tts1|recording.name=rec)
(t1| text2speech.name=tts2|rec|null)
(voting)
(tts2)
(i1)

The following compositions were identified by the algorithm, which also is implemented by Java according to the functional description presented in section 5.4.

- Composition1: null|information.name=i1|recognition|translation.name=t1
- Composition2: t1|text2speech.name=tts1|recording.name=rec
- Composition3: t1|text2speech.name=tts2|rec|null
- Composition4: voting
- Composition5: tts2
- Composition6: i1

As result the resolved identifiers for each keyword are presented in Table 7.3.

Table 7.3: Keywords and resolved identifiers

Keyword	Unique Name for Reuse	Identifiers
information	i1	information.greatwestern; information2.greatwestern
recognition		recogniser.att; recogniser.bt; recogniser.telefonica
Translation	t1	translation-en-de.leo
text2speech	tts1; tts2	tts.1und1; tts.alice
Recording	rec	record.google
Voting		vote.bt; vote.eplus; vote.o2

In accordance with the postulated features the algorithm determined 72 different solutions for the composition request, which were ordered regarding to their completeness (see Table B.0.1). Further tests have been defined and executed to validate the algorithm's functionality, as listed below.

- *(null|information|recognition)*
- *(information| null|recognition)*
- *(information|recognition|null)*
- *(information.name=i1|translation) (i1|text2speech)*
- *(conference|translation)*
- *(translation|text2speech.name=tts1|recording) (translation|tts1|recording)*
- *Multiplier*
- *null*

7.3.2 Validation of the framework functionality

For validating the framework different services were implemented and provided by three different application servers (see Figure 7.1). The following Table 7.4 gives an overview of the implemented services.

Table 7.4: Implemented Services

Keyword	SIP URI	Inputs	Outputs	Functionality	stand-alone
information	newsticker.eintracht	video(1)	video(2) text(1)*	news ticker	not supported
playback	music.mtv	none	video(1) music(1)	video on demand	-
text2speech	tts.oxford	text(1)	speech(1)	text to speech	impossible
information	karaoke.youtube	music(1) video(1)*	music(1) video(1)	subtitle	not supported
conference	audio.orange	speech(n)	speech(n)	audio conference	supported
conference	video.grundig	video(n)	video(n)	video conference	supported

*optional

Table 7.4 displays the implemented services with its keywords, by which they are searchable, SIP URIs, as unique identifiers and reachable addresses, input and output connectors characterised by their media types (see section 5.2), their internal functionalities and finally if they are non-stand-alone services which are enhanced to support utilisation in stand-alone mode as well.

All of these services are registered by the application servers at the framework and offered their service descriptions as already stated in sections 4.5.1 and 4.5.2. A SIP User Agent sends service composition requests to the framework. The framework compiles the resulting service compositions and returns the response for the service composition request to the SIP User Agent regarding to section 4.5.2. Consequently the SIP User Agent has the ability to choose which of the resulting sorted service composition he wants to request. After chosen one service composition it will be invoked as described in section 4.5.3. In the following all test scenarios for framework's functionality are listed.

- All services are requested and invoked as single services. The services with the SIP URIs sip:newsticker.eintracht@p2p.de, sip:tts.oxford@p2p.de and sip:karaoke.youtube@p2p.de rejected the call request as expected, if the

absence of a necessary input (see Table 7.4) is detected (e.g. SIP User Agent only offers audio within its session description).

- Simple composite services which are feasible, means maximum two services as a composite service. An overview of the service composition requests and the resulting responses can be seen in Appendix C from Table C.0.1.
- Simple grouping (aggregation) of services. This implies that the services inputs and outputs are not connected to each other and no service compositions are made. The following lists all possible aggregated service requests which are callable regarding to Table 7.4:
    - *(conference)(conference)*
    - *(conference)(playback)*
    - *(playback)(conference)*
    - *(conference)(conference)(playback)*
    - *(conference)(playback)(conference)*
    - *(playback)(conference)(conference)*
- Some complex service compositions with aggregation and reutilisation are validated. All of these which are tested are listed below.
    - *(conference\|information)(conference)*
    - *(conference\|information)(text2speech\|conference)*
    - *(playback\|information.name=i1)(i1\|text2speech\|conference)*
    - *(conference.name=c1)(playback\|information.name=i1)*
      (i1\|text2speech\|c1)
    - *(playback\|information\|conference)(information.name=i1\|conference)*
      (i1\|text2speech)

- *conference.name=c1|information|c1*
- *(conference.name=c1|information|c1)(c1|information)*

The resulting sets can be seen in Appendix C in Table C.0.2.

The utilised SIP User Agent implementation has the ability to support all necessary media types (audio, video, text). It also provides the definition of service composition requests and it can interpret the resulting responses. All tests had been performed manually.

7.4 Framework evaluation

Since the functionality of the framework has been validated, it has also to be evaluated. Regarding the criteria given in section 7.1 the framework has been evaluated (see Table 7.5).

Table 7.5: Framework Evaluation

Main Feature	Criterion
General Features	peer-to-peer infrastructure
	no central database
	no central coordinator
	no single point of failure
	data redundancy
	SIP utilisation
	support for existing services
	layered service concept
	support of arbitrary media
NGN key features	applicability for arbitrary services
	separation of call/service control and media data transport
	application server support
	support for multimedia services
	packet-based data transport
	scalability
	unrestricted access to different service providers
SOA features	loosely coupled services
	location independent services
	reusable services
	modular services
	composable services
	discoverability of services
	implementation independent service interfaces
	“find, bind and execute” -paradigm
Service Delivery Platform characteristics	support of service provisioning
	combine/create services from existing ones
	accessible through specified service interfaces
	hidden complexity by abstracted interfaces
	support of SOA concepts
SDL characteristics	machine interpretable
	human readable description
	platform and programming independent
	standardised language/technology
	functional and non-functional properties
	functionality characterised by keywords
	support for any human perceptual service

General Features

The framework's prototype was constructed on the basis of a SIP peer-to-peer infrastructure as described in section 4. Also there is no central database. All necessary data, location and service descriptions, are stored in a DHT-based database, which is accessible by the core elements, SIP Proxy, Registrar and discovery servers. Within the framework's prototype no central coordinator exists which controls the interaction of the peers among themselves. If one peer wants to contact another one it only utilises a SIP Proxy server for initial connection establishment (see section 2.2.1). Also if a service or a service combination will be called no central coordinator is involved (see section 4.5.3). Because of that all peers are SIP-based they include client and server functionalities according to (IETF RFC 3261, 2002). Furthermore the peers, the end users, are enabled to decide autonomously to accept or reject a call and they are also free to choose a suggested composition. If one central element is going offline, maybe caused by a failure, the stored data within the DHT is still available for the other elements; hence the criteria of no single point of failure and data redundancy are fulfilled. As long as not all central elements will break down at the same time the data is still available. To increase redundancy and stability more central elements can be implemented than done within the prototype. Due to the fact that the whole framework and also the prototype utilises SIP all standard SIP services and standardised SIP-based supplementary services are supported, e.g. Call Hold, Transfer and Automatic Redial (IETF RFC 5359, 2008). The prototype as well follows the layered service concept of telecommunication presented in section 3.2. The basis for support of arbitrary media has been set by the prototype. To enable fully support of arbitrary media further media types should be registered at IANA as MIME (Multipurpose Internet Mail Extensions)

Media Types; so they can be utilised within the SIP body by the SDP (IETF RFC 4566, 2006) and RTP as payload (IETF RFC 4855, 2007) for the media transport.

NGN key features

All of the given NGN key features are supported by the framework's prototype. Arbitrary services are applicable and provided by application servers. The application servers in combination with media servers support the criterion of support for multimedia services. The prototype implementation also follows the separation of call/service control and media data transport, because SIP and SDP is used for the call/service control and RTP for the media data transport. Furthermore the whole framework is based on IP, so the data transport is packet-based. And even the framework's prototype is highly scalable, because the restrictions for this are only given by the applied DHT implementation and hardware utilised for the prototype. The framework may make usage of another DHT implementation and any hardware. Unrestricted access to different service providers is fully supported, because there is no limitation regarding to service publishing and providing. Any service provider is enabled to register and publish their services at the framework.

SOA features

The services which are implemented within the prototype of the framework are following the SOA principles. So the services are loosely coupled, which means that they are not directly depending on each other. They can be more or less used separately. Also the services can be invoked from any location which has the ability to connect to the given IP infrastructure. The services are reusable, because they will open up a new instance each time they will be utilised. Furthermore the services are

modular, because composite services can be created by combining these services. All of the services within the prototype are discoverable by their service interfaces utilising the service discovery server. The service interfaces are implemented by the framework's SDL (see section 5.2) which is independent of the service implementation. And lastly the whole prototype has been implemented following the "find-bind-execute" paradigm (see section 4.5).

Service Delivery Platform characteristics

The support of the SOA concepts is provided, because all requested SOA features are supported. Service provisioning is also integrated into the prototype through the simple service registration process as described in section 4.5.1. Through the service composition server the creation of new services by reusing existing services is supported and implemented into the prototype. Due to the framework's SDL (see section 5.2) the service interface comprises a way to access the service by its interface description applying SIP URIs. Furthermore by applying the SDL and invocation concept through SIP URIs the complexity of the service is hidden through abstraction by the interface description.

SDL characteristics

All these characteristics already fulfilled by applying the SDL (see section 5.2). The framework's SDL is machine readable, has the possibility to add a human readable description of the service's functionality and it is also based on standardised language (here: XML) and thereby it is platform and programming language independent. Furthermore the SDL provides the possibility to describe functional and even non-functional properties. The functionality of the services are characterised by simple

keywords which are utilised for service and composition requests. And last but not least the possibility to describe any human perceptual service is an integral feature of the implemented SDL.

Furthermore the framework is evaluated according the reduction of the amount of signalling in comparison to NGSON. Figure 7.9 illustrates the steps to be taken to request a service composition in NGSON (IEEE Std 1903-2011, 2011).

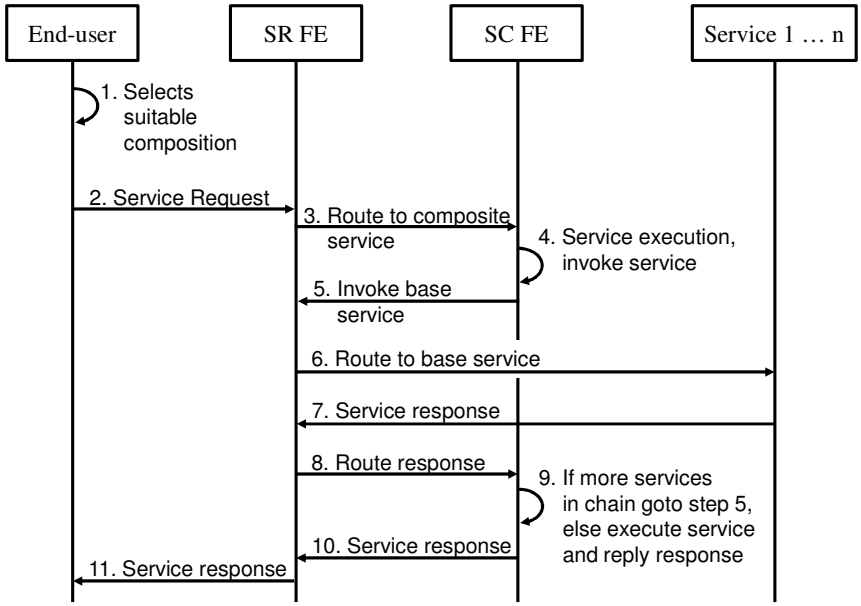


Figure 7.9: General service composition in NGSON

First a suitable service composition that has been requested before will be selected by the end-user. Next he sends out a service request to the service router functional entity (SR FE). This receives the request and forwards it to the service composition functional entity (SC FE). The SC FE then decides the order of service chaining (step 4). After that the SC FE requests invocation of a base service, which is one involved in service composition. SR FE receives the request and routes it to the base service provider. The service provider sends a response back to the SR FE. In step 8 the SR FE sends the received response to SC FE. Depending on the service request from step

3 the SC FE decides whether to invoke another base service (see step 5-8) based on the order of service chaining till the end of service chaining. If the service chain ended SC FE produces the final result of composite service and sends the result to the SR FE (see step 10). This result will then be forwarded to the end-user.

The steps 2 and 3, 5-8, and 10 and 11 presented in Figure 7.9 are related signalling. Assuming that the signalling during composite service request is also done by SIP in a NGSON implementation, it is comparable to the amount of signalling within the presented research work framework. Figure 7.10 illustrates the steps to be taken during a composite service request within the research work framework.

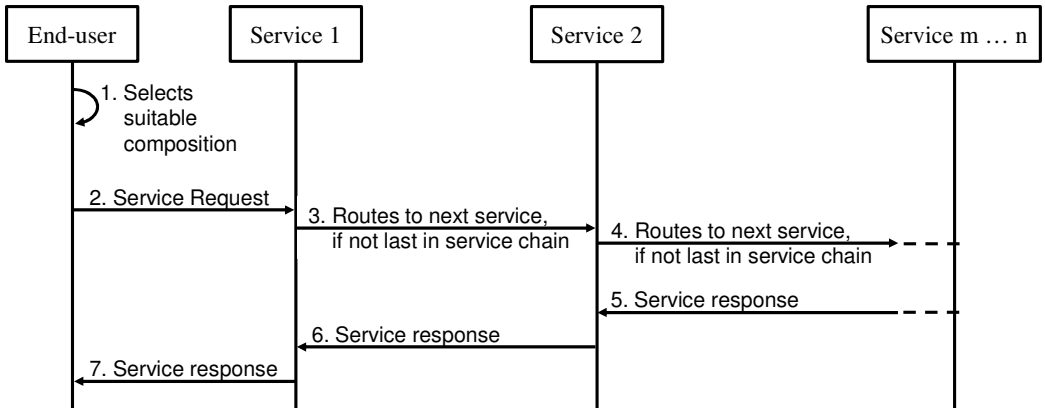


Figure 7.10: General service composition in framework

Here the end-user selects a suitable service composition, too. Then he directly sends his service request to the first service in composition chain, which routes the request to the next service in chain, if it is not the last in service chain. The following service behaves like the prior. Step 3 respectively step 4 repeat till end of service chain. Then the last service in chain sends it response to its ancestor until the first service within the service composition has been reached. This will then send the final response to the end- user (see step 7.). All steps in Figure 7.10 except step 1 are related to signalling.

The resulting amount of signalling messages (M) as a function of (n) services can be calculated as follows, regarding the research work framework.

$$M = 2n$$

Derived from the signalling behaviour in Figure 7.9 the following function results.

$$M = 2 + 2 + (4n)$$

End-user messages

SR - SC messages

Figure 7.11 illustrates the amount of messages as a function of services comparing NGSON and the research framework. It is easy to see that the amount of signalling messages increases much more regarding to the number of services in NGSON than in the research framework.

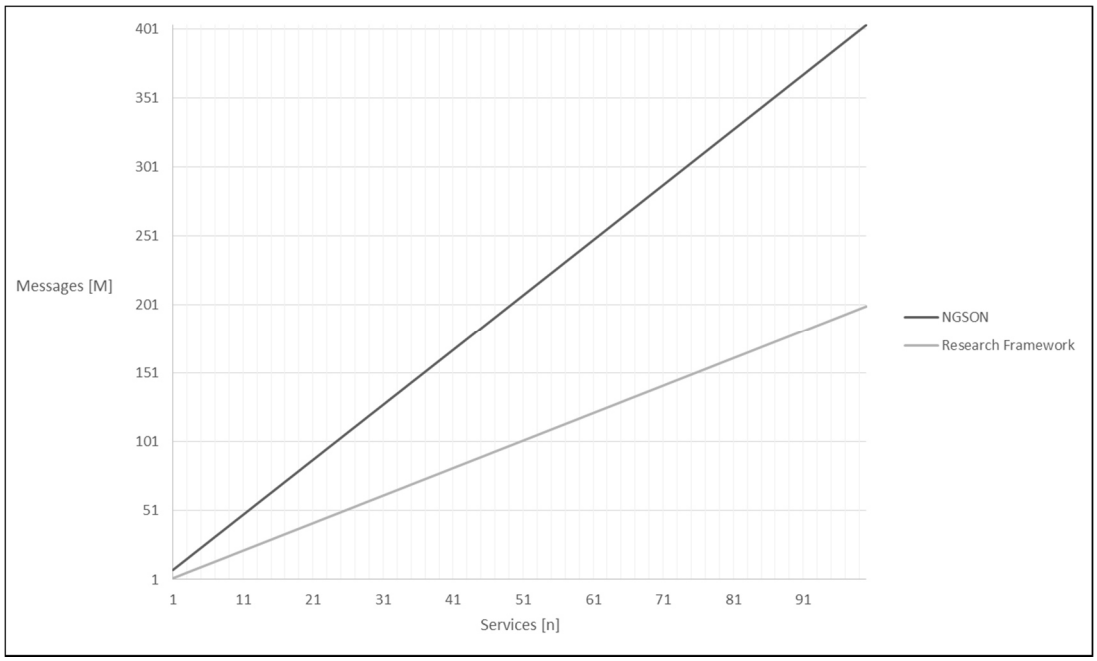


Figure 7.11: Comparison diagram

The slope of the function of services is two times bigger in NGSON. In conclusion, the efficiency regarding signalling for composite services of the research framework is two times better than in NGSON.

7.5 Summary

Within this chapter, the research prototype developed within this project has been introduced. Its architecture and the way it has been implemented are outlined. Also the functionalities of the framework's prototype are depicted. The validation process of the research prototype is presented by applying different service request scenarios. The research prototype has been successfully implemented for a proof of concept evaluation of the proposed framework, demonstrating its functionalities and its general applicability. The research prototype and the proposed framework have been evaluated conclusively and it could be shown that all criteria for evaluation are fulfilled by the framework.

8 Conclusions

This chapter concludes the thesis by summarising the achievements of the research work (section 8.1). Also the limitations of the accomplished research work are presented (section 8.2), and scopes for further research are suggested (section 8.3).

8.1 Achievements of the research

The research work of this thesis was dedicated to the development of a novel approach for service composition, discovery and provisioning within SIP-based peer-to-peer networks involving the concepts of NGN and SOA. A complete framework was defined, completed by the elaboration of essential detail aspects and the introduction of novel mechanisms. The framework can be applied as a fully-fledged solution for service provisioning, discovery and composition following the concepts and principles regarding to services from NGN and SOA utilising SIP.

The analysis of existing approaches concerning service composition concepts (see section 3.4) clearly illustrated the necessity for an optimised SIP-based service composition concept. Furthermore the support of a simple and flexible way to provide new services within SIP-based peer-to-peer networks had been clarified (see section 4.5.1), in relation to service provisioning in IMS (see section 3.3), which is a centralised approach of a NGN. Also the essential concepts of service in SOA and telecommunication were highlighted and applied for comparison of the existing service composition concepts for NGN (see sections 3.1 and 3.2). The results of chapter 3 shaped the general requirements the framework is based on.

Applying these requirements, a novel framework has been engineered, providing the derived functionalities from SOA for service discovery, binding and publishing as well as service composition (see chapter 4). The framework architecture consists of redundant SIP Proxy and Registrar servers utilising a distributed location service which is based on a DHT. These SIP servers represent the service stratum of NGN and act like super nodes of a peer-to-peer network. In addition two further server functionalities had been introduced to fulfil the need of service discovery, publishing and composition; these functionalities belong to the application stratum of NGN. The discovery and composition functionalities, which are integral elements of the SIP servers, have access to a distributed repository storing all the service descriptions. Among others the required mechanisms based on SIP for the provisioning, discovery and composition have been defined and illustrated.

The basis of service composition and discovery for this framework has been developed in chapter 5. Here a completely new service description language has been defined. Previously different service description languages have been discussed and compared afterwards with the novel description language. The novel service description language satisfies all requested characteristics of which several were derived from the SOA principles. Also an adequate algorithm has been designed which enables the framework to create service compositions by utilising information from the service descriptions. The algorithm will be initiated by a composition request, which is built upon a simple syntax to describe requests by keywords.

Further recommendations regarding the service creation by developers have been made (see chapter 6). These recommendations have been defined to reduce the risk

of unwanted service interactions. The framework provides mechanisms, which have been illustrated, to implement the presented recommendations.

For the evaluation of the overall framework functionalities, a research prototype has been implemented, providing all relevant framework functionalities (see chapter 7). The framework functionalities of the prototype have been developed in Java utilised open source libraries. To be as near as possible to a real world implementation the prototype has been setup on Linux virtual machines connected together by open source network simulator GNS3. Furthermore services have been implemented whereby the functionality of service composition has been validated. Through these verifications it has been shown, that the framework fulfils all requirements.

In general, the proposed framework is an effective solution for service delivery platform integration into a SIP-based NGN. The framework has a number of advantages in contrast to IMS with respect to service provisioning, discovery and composition, which are that all services are discoverable through an open service description interface, dynamic service compositions can be realised by the framework without the need to extend it with further elements such as SCIM or Service Brokers or even to link an overlay such as NGSON and finally that services can be published and provided easily using SIP, so they can be provided spontaneously and no special OSS is needed. A number of publications to different aspects of this research have been presented at conferences and have received positive feedback from reviewers and attendants.

8.2 Limitations of the research

Even though the objectives of the research project have been met, some limitations exist, because of decisions which had to be made. Those decisions were caused by practical reasons and were made to delimit the range of related fields concerning to aspects such as QoS or billing in the context of services and telecommunication. The main limitations are listed below.

1. The presented framework is based on a partially centralised architecture. Therefore a peer-to-peer network with so-called super nodes has been chosen. These super nodes are elements with extended functionalities, such as service composition, than ordinary nodes. Depending on the point of view this aspect may be an advantage or disadvantage. Anyway, the framework can easily be modified to be fully decentralised or implemented as a centralised solution.
2. The framework lacks of a definition for a responsible entity for the definition of the keywords, which have to be used to describe the service functionalities. This depends on the way the framework is implemented. If the framework is implemented by a provider the responsibility for the keywords should be given to him, in order to avoid misunderstandings and to ensure the framework's functionality to create service compositions.
3. The research prototype designed for this project has been implemented to provide all essential functionalities of the framework to allow a general proof of concept. However, the prototype has not been finalised regarding the algorithm implementation. The sort of the resulting compositions by its specified non-functional properties, e.g. QoS parameters, has not been

implemented. While this is not an essential functionality of the framework, it is easily feasible to be implemented into the algorithm.

Despite these limitations, the research project has made valid contributions to knowledge and provided sufficient proof of concept for the proposed approaches.

8.3 Suggestions and scope for Future Work

This research project has advanced the field of service provisioning and composition for SIP-based peer-to-peer networks. Nevertheless, several areas for future work have been identified during the course of the research, as enhancements for this project. Some of these already have been mentioned, however they are summarised below.

An extension to the presented framework could be defined, allowing for the full decentralisation of the whole network architecture, so that all peers are equal. Therefore the functionalities of the SIP Proxy, Registrar, and discovery and composition server have to be included within every peer. Furthermore the framework could be extended by the functionality of detecting service interactions in live operation and resolve these interactions during service invocation. For this purpose extensive research is required. Even further research work on focusing to enable the customers to search more precisely for service compositions could be arranged. This might lead to support of natural language support as an input for service and service composition requests, which also requires extensive research.

References

1. 3GPP TS 23.228 V5.15.0 (2006), Technical Specification, “IP Multimedia Subsystem (IMS); Stage 2 (Release 5)”, 3GPP
2. 3GPP TS 23.218 V6.4.0 (2006), Technical Specification, “IP Multimedia session handling; IM call model”, 3GPP
3. 3GPP TR 23.810 V1.0.0 (2008), Technical Report, “Study on Architecture Impacts of Service Brokering”, 3GPP
4. Allen, J. F. (1983), “Maintaining knowledge about temporal intervals”, Magazine Communications of the ACM, Volume 26, Issue 11, pp. 832-843, ACM
5. Amyot, D. and Logrippo, L. (2003), “Feature Interactions in Telecommunications and Software Systems VII”, IOS Press, Fairfax, USA, ISBN: 1-58603-348-4
6. Bether, C. (2008), “Software Service Composition in Next Generation Networking Environments”, Dissertation, Technische Universität (TU) Berlin, Germany
7. Bischof, Udo (2005), “Strukturierte P2P Netze”, Seminar P2P Networking, Brandenburgische Technische Universität (BTU) Cottbus – Brandenburgian Technical University Cottbus, Germany
8. Blum, N.; Magedanz, T.; Schreiner, F. (2009), “Management of SOA based NGN service exposure, service discovery and service composition”, International Symposium on Integrated Network Management, pp.430-437, IEEE
9. Boles, D.; Becker, A.; Bley, S.; Dauelsberg, M.; Esser, A.; Knoblich, C.; Kölling, M.; Logemann, D.; Mertins, G.; Prusch, T.; Stehen, B.; Unbehaun, S.; Voßkamp, R. (1996), “Zwischenbericht der Projektgruppe Multimedia-Präsentation im Gesundheitswesen Teil B”, (translated title: “Progress report of the project group multimedia presentation in public health sector”), University Oldenburg, Germany
10. Bouma, W. and Velthuijsen, H. (1994), “Feature Interactions in Telecommunications Systems”, IOS Press, Fairfax, USA, ISBN: 90-5199-165-7
11. Bousquet, L. and Richier, J.-L. (2007), “Feature Interactions in Software and Communication Systems IX”, IOS Press, Fairfax, USA, ISBN: 1-58603-845-8
12. Branca, G. and Atzori, L. (2012), “A Survey of SOA Technologies in NGN Network Architectures”, Communications Surveys & Tutorials Magazine, Volume 14, Issue 3, pp. 644-661, IEEE

13. Bravo, O.; Costa, A.; Nicolau, M.J. (2012), "Design and implementation of a hierarchical SIP-based peer-to-peer network", 20th International Conference on Software, Telecommunications and Computer Networks, pp. 1-9, IEEE

14. BrightStar – Enterprise Consulting (SAP|Oracle) (2014), "Comparison of SOA Suites", Available at: http://bslion.blog.com/2012/12/18/comparison-of-soa-suites/, [accessed 24th July 2014]

15. British Telecom (2008), "Web21CSDK", Available at: http://web21c.bt.com, [accessed 28th June 2008]

16. Calder, M. and Magill, E. H. (2000), "Feature Interactions in Telecommunications and Software Systems VI", IOS Press, Fairfax, USA, ISBN: 1-58603-065-5

17. Calder, M.; Kolberg, M.; Mahill, E. H.; Reiff-Marganiec, S. (2003), "Feature interaction: a critical review and considered forecast", The International Journal of Computer and Telecommunications Networking, Volume 41, Issue 1, pp. 115-141, Elsevier

18. Cheng, K. E. and Ohta, T. (1995), "Feature Interactions in Telecommunications Systems III", IOS Press, Fairfax, USA, ISBN: 90-5199-238-6

19. Cheung, E. and Purdy, K.H. (2008), "An Application Router for SIP Servlet Application Composition", International Conference on Communications (ICC 2008), pp. 1802-1806, IEEE

20. Chinnici, R.; Moreau, J.-J.; Ryman, A.; Weerawarana, S. (2007), "Web Services Description Language (WSDL) Version 2.0 Part1: Core Language", W3C Recommendation

21. Cochennec, Jean-Yves (2002), "Activities on next-generation networks under Global Information Infrastructure in ITU-T", Communication Magazine, Volume 40, Issue 7, pp.98-101, IEEE

22. Cordier, C.; Carrez, F.; Van Kranenburg, H.; Licciardi, C.; Van der Meer, J.; Spedalieri, A.; Le Rouzic, J.-P. (2006), "Addressing the challenges of Beyond 3G service delivery: the SPICE service platform", Workshop on Applications and Services in Wireless Networks (ASWN'2006), Berlin, Germany

23. Cormen, T.H.; Leiserson, Ch.E.; Rivest, R.L.;Stein, C. (2009), "Introduction to Algorithms (3rd Edition)", MIT Press, Cambridge, USA, ISBN:978-0-262-53305-8

24. Deutsche Telekom (2014), "Developer Garden", Available at: http://www.developergarden.com, [accessed 25th February 2014]

25. Dini, P.; Boutaba, R.; Logrippo, L. (1997), "Feature Interactions in Telecommunications Networks IV", IOS Press, Fairfax, USA, ISBN: 90-5199-347-1

26. Distributed and Mobile Systems Group of Bamberg University (2008), "Open Chord", Available at: http://open-chord.sourceforge.net/, [accessed 2nd October 2013]

27. Eisenschmidt, Wilhelm (2005), “Peer-to-Peer-Architekturen”, Proseminar Virtuelle Präsenz, University Ulm, Germany

28. Erl, Thomas (2007), “SOA: Principles of Service Design”, Prentice Hall, Inc., New Jersey, USA, ISBN: 978-0132344821

29. ETSI DTR 01024 V0.1.0 (2005), Draft Technical Report, “NGN generic capabilities and their use to develop services”, ETSI TISPAN

30. ETSI Tdoc RP 030375 V0.10 (2003), Technical Document, “Overview of 3GPP Release 5”, ETSI Mobile Competence Centre

31. ETSI TS 122.001 V8.0.0 (2009), Technical Specification, “Principles of circuit telecommunication services supported by a Public Land Mobile Network (PLMN)”, ETSI

32. ETSI TS 122.101 V9.3.0 (2009), Technical Specification, “Service aspects; Service principles”, ETSI

33. ETSI TS 122.105 V8.4.0 (2008), Technical Specification, “Services and service capabilities”, ETSI

34. ETSI TS 122.127 V9.0.0 (2010), Technical Specification, “Service requirement for the Open Services Access (OSA)”, ETSI

35. ETSI TS 122.228 V8.6.0 (2009), Technical Specification, “Service requirements for the Internet Protocol (IP) multimedia core network subsystem (IMS)”, ETSI

36. ETSI TS 123.198 V9.0.0 (2010), Technical Specification, “Open Service Access (OSA)”, ETSI

37. ETSI TR 180 000 V1.1.1 (2006), Technical Report, “NGN Terminology”, ETSI TISPAN

38. ETSI TR 188 001 V1.2.1 (2006), Technical Report, “Operation Support Systems Architecture”, ETSI TISPAN

39. Eurescom EDIN 0531-1652 (2006), Technical Information, “Service Oriented Architecture and Telcos”, Eurescom

40. Euzénat, J.; Ferrara, A.; Hollink, L.; Isaac, A.; Joslyn, C.; Malaisé, V.; Meilicke, Ch.; Nikolov, A.; Pane, J.; Sabou, M.; Scharffe, F.; Shvaiko, P.; Spiliopoulos, V.; Stuckenschmidt, H.; Sváb-Zamazal, O.; Svátek, V.; Trojahn dos Santos, C.; Vouros, G.; Wang, S. (2009), “Results of the Ontology Alignment Evaluation Initiative 2009”, Proceedings of the 4th international workshop on Ontology Matching (OM-2009), pp. 73-126, CEUR-WS

41. Falconer, S. M. and Noy, N. F. (2011), “Schema Matching and Mapping - Interactive Techniques to Support Ontology Matching”, pp. 29-51, Springer-Verlag, Berlin Heidelberg, Germany, ISBN: 978-3-642-16517-7

42. Garces-Erice, L; Felber, P.A.; Biersack, E.W.; Urvoy-Keller, G.; Ross, K.W. (2004), “Data indexing in peer-to-peer DHT networks”, Proceedings of the 24th

International Conference on Distributed Computing Systems, pp. 200-208, IEEE

43. GNS3 (2013), "GNS3", Available at: http://www.gns3.net, [accessed 1st October 2013]

44. Gouya, A.; Crespi, N.; Bertin, E. (2006), "SCIM (Service Capability Interaction Manager) Implementation Issues in IMS Service Architecture", International Conference on Communication (ICC 2006), Volume 4, pp. 1748-1753, IEEE

45. Gstreamer (2013), "gstreamer open source multimedia framework", Available at: http://gstreamer.freedesktop.org, [accessed 4th October 2013]

46. Harjula, Erkki; Ylianttila, Mika; Ala-Kurikka, Jussi; Riekki, Jukka; Sauvola, Jaakko (2004), "Plug-and-Play Application Platform: Towards Mobile Peer-to-Peer", Proceedings of the 3rd International conference on Mobile and Ubiquitous multimedia (MUM2004),pp. 63-69, ACM Press, USA, ISBN: 1-58113-981-0

47. Hashimi, S. (2003), "Service-Oriented Architecture Explained", Available at: http://windowsdevcenter.com/pub/a/dotnet/2003/08/18/soa_explained.html, [accessed 17th December 2012] O'Reilly Media, Inc., Sebastopol, USA

48. Harte, J. Lawrence; Hoenig, M; McLaughlin, D.; Kikta, R. (1999), "CDMA IS-95 for Cellular and PCs: Technology, Economics, and Services", McGraw-Hill, New York, USA, ISBN: 0-07-027070-8

49. Hewitt, E: (2009), "Java SOA Cookbook", O'Reilly Media, Inc., Sebastopol, USA, ISBN: 978-0-596-52072-4

50. IEEE P1903/D1 (2008), "Draft White Paper for Next Generation Service Overlay Network", IEEE

51. IEEE P1903.1 (2014), "Standard for Content Delivery Protocols of Next Generation Service Overlay Network (NGSON)", Available at: http://standards.ieee.org/develop/project/1903.1.html, [accessed 29th July 2014], IEEE

52. IEEE P1903.2 (2014), "Standard for Service Composition Protocols of Next Generation Service Overlay Network (NGSON)", Available at: http://standards.ieee.org/develop/project/1903.2.html, [accessed 29th July 2014], IEEE

53. IEEE P1903.1 (2014), "Standard for Self-Organizing Management Protocols of Next Generation Service Overlay Network (NGSON)", Available at: http://standards.ieee.org/develop/project/1903.3.html, [accessed 29th July 2014], IEEE

54. IEEE Std 1903-2011 (2011), "IEEE Standard for the Functional Architecture of Next Generation Service Overlay Networks", IEEE

55. IETF RFC 1034 (1987), Request For Comments, "DOMAIN NAMES – CONCEPTS AND FACILITIES", IETF

56. IETF RFC 1035 (1987), Request For Comments, "DOMAIN NAMES – IMPLEMENTATION AND SPECIFICATION", IETF

57. IETF RFC 2136 (1997), Request For Comments, “Dynamic Updates in the Domain Name System (DNS UPDATE)”, IETF

58. IETF RFC 3261 (2002), Request For Comments, “SIP: Session Initiation Protocol”, IETF

59. IETF RFC 3264 (2002), Request For Comments, “An Offer/Answer Model with the Session Description Protocol (SDP)”, IETF

60. IETF RFC 3265 (2002), Request For Comments, “Session Initiation Protocol (SIP) Specific Event Notification”, IETF

61. IETF RFC 3480 (2004), Request For Comments, “Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)” IETF

62. IETF RFC 3550 (2003), Request For Comments, “RTP: A Transport Protocol for Real-Time Applications”, IETF

63. IETF RFC 3903 (2004), Request For Comments, “Session Initiation Protocol (SIP) Extension for Event State Publication”, IETF

64. IETF RFC 4234 (2005), Request For Comments, “Augmented BNF for Syntax Specifications: ABNF”, IETF

65. IETF RFC 4566 (2006), Request For Comments, “SDP: Session Description Protocol”, IETF

66. IETF RFC 4855 (2007), Request For Comments, “Media Type Registration of RTP Payload Formats”, IETF

67. IETF RFC 5359 (2008), Request For Comments, “Session Initiation Protocol Service Examples”, IETF

68. IETF RFC 6733 (2012), Request For Comments, “Diameter Base Protocol”, IETF

69. IETF RFC 6940 (2014), Request For Comments, “Resource Location And Discovery (RELOAD) Base Protocol”, IETF

70. ISC (2013), “The most widely used name server software: BIND”, Available at: https://www.isc.org/downloads/bind/, [accessed 2nd October 2013], Internet Systems Consortium

71. ITU-T I.210 (1993), Recommendation, “Principles of Telecommunication Services supported by an ISDN and the means to describe them”, ITU-T

72. ITU-T I.211 (1993), Recommendation, “B-ISDN Service aspects”, ITU-T

73. ITU-T Y.2001 (2004), Recommendation, “General overview of NGN”, ITU-T

74. ITU-T Y.2012 (2010), Recommendation, “Functional requirements and architecture of next generation networks”, ITU-T

75. ITU-T Y.2234 (2008), Recommendation, “Open service environment capabilities for NGN”, ITU-T

76. ITU-T M.3060/Y.2401 (2006), Recommendation, “Principles for the Management of Next Generation Networks”, ITU-T

77. Jackson, M. and Zave, P. (1998), “Distributed Feature Composition: A Virtual Architecture for Telecommunications Services”, IEEE Transactions on Software Engineering Journal, Volume: 24, Issue: 10, pp. 831-847, IEEE

78. Jennings, C.; Lowekamp, B.; Rescola, E.; Baset, S.; Schulzrinne, H. (2012), “Resource Location And Discovery (RELOAD) Base Protocol”, Internet-Draft Version 23 (Expires May 9, 2013), IETF

79. Josuttis, N.M. (2007), “SOA in Practice: The Art of Distributed System Design (1st Edition)”, O’Reilly Media Inc., Sebastopol, USA, ISBN: 978-0-596-52955-0

80. JSDL (2014), “JSDL – JSON Service Description Language”, Available at: https://jsdl.codeplex.com, [accessed 30th July 2014], Mocrosoft

81. JSR 289 (2008), Java Specification Request, “SIP Servlet Specification, version 1.1”, JCP

82. Kalasapur, S.; Kumar, M.; Shirazi, B. A. (2007), “Dynamic Service Composition in Pervasive Computing”, Journal IEEE Transactions on Parallel and Distributed Systems, Volume 18, Issue 7, pp. 907-918, IEEE

83. Kim, S. and Jeong, J. (2013), “Design and Performance Analysis of A Novel P2P-SIP Architecture for Network-Based Mobility Support in Intelligent Home Networks”, IEEE Ninth International Conference on Mobile Ad-hoc and Sensor Networks, pp. 310-317, IEEE

84. Kimbler, K. and Bouma, W. (1998), “Feature Interactions in Telecommunications and Software Systems V”, IOS Press, Fairfax, USA, ISBN: 90-5199-431-1

85. Kolberg, M. and Magill, E.H. (2001), “Handling incompatibilities between services deployed on IP-based networks”, Intelligent Network Workshop, pp. 360-370, IEEE

86. Krafzig, D.; Banke, K.; Slama, D. (2005), “Enterprise SOA: Service-Oriented Architecture Best Practices”, Pearson Education, Inc., New Jersey, USA, ISBN: 0-13-146575-9

87. Kühn, P. (1991), “Vorlesungsskript Nachrichtenvermittlung I und II” (translated title: “Lecture notes message switching I and II”), University Stuttgart, Institut für Nachrichtentvermittlung und Datenverarbeitung

88. Kwok, T. (1995), “A vision for residential broadband services: ATM-to-the home”, IEEE Network Magazine, Volume 9, Issue 5, pp. 14-28, IEEE

89. Lee, Seung-Ik; Kang, Shin-Gak (2012), “NGSON: features, state of the art, and realization”, IEEE Communications Magazine, Volume 50, Issue 1, pp. 54-61, IEEE

90. Lehmann, Armin; Trick, Ulrich (2007), “Abschlussbericht Detecon – Services in NGN” (translated title: “Final Report Detecon – Services in NGN”),

University of Applied Sciences Frankfurt am Main and Detecon International GmbH

91. Lehmann, Armin; Eichelmann, Thomas; Trick, Ulrich (2008a), "Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation" (translated title: "New possibilities for service provisioning based on Peer-to-Peer communication"), 13. VDE/ITG (Verband der Elektrotechnik Elektronik Informationstechnik/Informationstechnische Gesellschaft) Mobilfunktagung Osnabrück, Germany

92. Lehmann, Armin; Fuhrmann, Woldemar; Trick, Ulrich; Ghita, Bogdan (2008b), "New possibilities for the provision of value-added services in SIP-based peer-to-peer networks", Proceedings of SEIN (Symposium on Security, E-learning, Internet and Networking), University of Wrexham, United Kingdom

93. Lehmann, A; Trick, U; Fuhrmann, W. (2009a), "SOA-basierte Peer-to-Peer-Mehrwertdienstebereitstellung" (translated title: "SOA based peer-to-peer provisioning of value-added services"), 14. VDE/ITG (Verband der Elektrotechnik Elektronik Informationstechnik/Informationstechnische Gesellschaft) Mobilfunktagung Osnabrück, Germany

94. Lehmann, Armin; Fuhrmann, Woldemar; Trick, Ulrich; Ghita, Bogdan (2009b), "A new approach to classify and describe telecommunication services", Proceedings of SEIN (Symposium on Security, E-learning, Internet and Networking), University of Applied Sciences Darmstadt, Germany

95. Lehmann, A.; Trick, U.; Fuhrmann, W.; Ghita, B. (2011), "A new service description language as basis for service composition in SIP-based peer-to-peer infrastructures", 4th Wireless and Mobile Networking Conference (WMNC), pp. 1-8, IEEE

96. Leon Trisha (2014), "Telecom And IT Convergence Creating Opportunities For VARs", Available at: http://www.bsinfo.com/doc/telecom-and-it-convergence-creating-opportunities-for-vars-0001, [accessed 31st July 2014], Jameson Publishing

97. Lennox, J. and Schulzrinne, H. (2000), "Feature Interaction in Internet Telephony", 6th Feature Interaction Workshop, Glasgow, United Kingdom

98. Lu, H.; Zheng, Y.; Sun, Y. (2008), "The Next Generation SDP Architecture: Based on SOA and Integrated with IMS", Second International Symposium on Intelligent Information Technology Application (IITA'08), Volume 3, pp. 141-145, IEEE

99. Magedanz, T.; Blum, N.; Dutkowski, S. (2007), "Evolution of SOA Concepts in Telecommunications", Computer Magazine, Volume 40, Issue 11, pp.46-50, IEEE

100. Makaya, C.; Falchuk, B.; Chee, D.; Lin, F.J.; Das, S.; Ito, M.; Komorita, S.; Chiba, T; Yokota, H. (2011), "Service Composition Based on Next-Generation Service Overlay Networks Architecture", 4th International Conference on New Technologies, Mobility and Security (NTMS), pp. 1-6, IEEE

101. Martin, D.; Burstein, M.; Hobbs, J.; Lassila, O.; McDermott, D.; McIlraith, S.; Narayanan, S.; Paolucci, M.; Parsia, B.; Payne, T; Sirin, E.; Srinivasan, N.; Sycara, K. (2004), "OWL-S: Semantic Markup for Web Services", W3C Member Submission, W3C

102. Metrics2 (2014), "Eclipse Metrics plugin 1.3.8 – Getting started", Available at: http://metrics2.sourceforge.net, [accessed 1st March 2014]

103. Mika, P.; Oberle, D.; Gangemi, A.; Sabou, M. (2004), Foundations for service ontologies: aligning OWL-S to dolce", Proceedings of the 13th international conference on World Wide Web (WWW '04), pp. 563-572, ACM

104. MobileIN (2004), "Value-added Services", Available at: http://www.mobilein.com/what_is_a_VAS.htm, [accessed 18th December 2012], MobileIN

105. Moriana Group (2013), "Moriana on SDP 2.0: Service Delivery Platforms – definition and evolution", Technology Article, Available at: http://www.morianagroup.com/index.php?option=com_content&view=article&id=148&Itemid=233, [accessed 29th April 2013], The Moriana Group

106. MSF IA SIP 005 (2004), "Implementation Agreement for SIP interface between Call Agent and Service Broker", MSF

107. MSF IA SIP 006 (2005), "Implementation Agreement for SIP interface between Service Broker and Application Server", MSF

108. Nakamura, M. and Reiff-Marganiec, S. (2009), "Feature Interactions in Software and Communication Systems X", IOS Press, Fairfax, USA, ISBN: 978-1-60750-014-8

109. OASIS Standard (2006), "Reference Model for Service Oriented Architecture 1.0", OASIS

110. OMA ERD OWSER 2006, Enabler Release Definition, "Enabler Release Definition for OMA Web Services Enabler (OWSER)", OMA

111. OMA RRD OSE 2009, Reference Release Definition, "Reference Release Definition for Open Mobile Alliance Service Environment (OSE)", OMA

112. OpenJSIP (2013), "OpenJSIP Open Java SIP – opensource SIP services implemented in Java (SIP Proxy, SIP Registrar etc.)", Available at: https://code.google.com/p/openjsip/, [accessed 2nd October 2013]

113. Oracle (2013), "VirtualBox", Available at: https://www.virtualbox.org, [accessed 1st October 2013]

114. Orange Partner (2014), "Orange Partner", Available at: http://www.orangepartners.com/, [accessed 25th February 2014]

115. Otto, W. (2005), "Fix zu Diensten" (translated title: "Fixing of services"), Funkschau 17/2005, Weka-Fachzeitschriften Verlag, Poing, Germany

116. Parlay 4.0 (2004), „Parlay X Web Services Specification – Version 1.0.1", Parlay Group, Inc.

117. Petruhu, C. (2014), "Envisioning Converged Service Delivery Platforms (SDP 2.0) – Part II", Service Technology Magazine, Issue LXXXII, March 2014, Prentice Hall

118. Poikselkä, Miikka and Mayer, Georg (2009), "IMS, IP Multimedia Concepts and Services (3rd edition)", John Wiley & Sons, Chichester, United Kingdom, ISBN:978-0-470-72196-4

119. Pu, Fan (2006), "P2P architecture for IP telephony using SIP", Mobile Communities Seminar on Internetworking, Helsinki University of Technology, Finland

120. Rahman, Mahfuzur and Buford, John (2006), Matsushita Electric Industrial Co., LTD. Service Discovery using Session Initiation Protocol. Int. Pat. WO 2006/060375 A2

121. Reiff-Marganiec, S. and Ryan, M. (2005), "Feature Interactions in Telecommunications and Software Systems VIII", IOS Press, Fairfax, USA, ISBN: 1-58603-524-X

122. Ritchie, D. M. (1984), "The Evolution of the Unix Time-sharing System", AT&T Bell Laboratories Technical Journal 63 No. 6 Part 2, pp.1577-93, Bell Laboratories New Jersey, USA

123. Roach, A.; Jennings, C.; Peterson, J.; Barnes, M. (2013), Session Initiation Protocol Parameters - Option Tags, Available at: http://www.iana.org/assignments/sip-parameters/sip-parameters.xml#sip-parameters-4. [accessed 5th March 2013]

124. Robie, J; Cavicchio, R.; Sinnema, R.; Wilde, E. (2013), „RESTful Service Description Language (RSDL): Describing RESTful Services Without Tight Coupling", Balisage Series on Markup Technologies, vol.10, The Markup Conference 2013, Montreal, Canada

125. Rodriguez, A. (2008), "RESTful Web services: The basics", IBM developerWorks, Available at: http://www.ibm.com/developerworks/webservices/library/ws-restful/, [accessed 23rd April 2013]

126. Rosen, Michael; Lublinsky, Boris; Smith, Kevin T.; Balcer, Marc J. (2008), "Applied SOA: Service-Oriented Architecture and Design Strategies", Wiley Publishing, Inc., Indianapolis, USA, ISBN: 978-0-470-22365-9

127. Salisbury, George (2006), "Trends in telecommunications services", Detecon International GmbH, Available at: http://www.detecon-dmr.com/en/article/trends-in-telecommunications-services_2006_12_31 [5th November 2013]

128. Salisbury, George (2007), "NGN and IMS: In and around regulation", Opinion Paper, Detecon International GmbH

129. Schmidt, H.; Guenkoca-Luy, T.; Hauck, F.J. (2006), "Service Location using the Session Initiation Protocol (SIP)", International conference on Networking and Services (ICNS), page 60, IEEE

130. Seidel, A. (2011), “Aufbau einer SIP-basierten Peer-to-Peer-Infrastruktur zur Bereitstellung von kombinierbaren Mehrwertdiensten” (translated title: “Construction of a SIP-based peer-to-peer infrastructure for provisioning of combinable value-added services”), Diploma Thesis, University of Applied Sciences Frankfurt am Main, Germany

131. Shingledecker, R. (2008), “Welcome to The Core Project – Tiny Core Linux”, Available at: http://www.tinycorelinux.net, [accessed 2nd October 2013]

132. Silva, E.; Pires, L.; van Sinderen, M. (2007), “An Algorithm for Automatic Service Composition”, 1st International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing (ICSOFT’2007), Barcelona, Spain, pp.65-74, INSTICC Press, ISBN: 978-989-8111-08-1

133. Singh, K. N. (2006), “Reliable, Scalable and Interoperable Internet Telephony”, PhD thesis, School of Arts and Sciences, Columbia University, USA

134. Singh, Kundan (2010), “Why do P2P-SIP?”, Available at: http://p2p-sip.blogspot.de/2010/10/why-do-p2p-sip.html, [accessed 31st July 2014]

135. Singh, Kundan and Schulzrinne Henning (2004), “Peer-to-Peer Internet Telephony using SIP”, Technical Report CUCS-044-04, Columbia University, USA

136. Singh, Kundan and Schulzrinne Henning (2006), “Using an External DHT as a SIP Location Service”, Technical Report CUCS-007-06, Columbia University, USA

137. Steinmetz, R. (2000), “Multimedia-Technologie: Grundlagen, Komponenten und Systeme (3rd edition)”, Springer, Berlin, Germany, ISBN: 3-540-67332-6

138. Stoica, I.; Morris, R.; Karger, D.; Kaashoek, M. F.; Balakrishnan, H. (2001), “Chord: A scalable peer-to-peer lookup service for internet applications”, Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM ’01), pp. 149-160, ACM, New York, USA, ISBN: 1-58113-411-8

139. TINA-C Deliverable (1997), “Service Architecture Version 5.0” TINA-C

140. Townsend, R. (2009), “An enabler for an emerging vision for future service networks”, Joint Workshop on Overlay Networking, Presentation, ITU

141. Trossen, D. and Pavel, D. (2005), “Service discovery & availability subscriptions using SIP event framework”, International Conference on Communication (ICC 2005), Vol. 3, pp. 1572-1577, IEEE

142. Trick, Ulrich and Weber, Frank (2004), “SIP, TCP/IP und Telekommunikationsnetze (1st edition)”, Oldenbourg, Munich, Germany, ISBN: 3-486-27529-1

143. Trick, Ulrich and Weber, Frank (2009), “SIP, TCP/IP und Telekommunikationsnetze (4th edition)”, Oldenbourg, Munich, Germany, ISBN: 3-486-59000-5

144. T-Systems (2007), “White Paper Next Generation Network: Motivation and Challenges for Incumbents”, T-System Enterprise Services GmbH, 2007

145. UNSPSC (2013), “UNSPSC”, Available at: http://www.unspsc.org, [accessed 14th March 2013], UNSPSC

146. Velez, J. and Correia, M. (2000), “Classification and Characterisation of Mobile Broadband Services”, 52nd Vehicular Technology Conference (VTC 2000), Volume 3, pp. 1417-1423, IEEE

147. WADL (2009), “Web Application Description Language”, Available at: http://www.w3.org/Submission/wadl/, [accessed 30th July 2014], W3C

148. Weber, Frank (2012), “Quality of Service optimisation framework for Next Generation Networks”, PhD thesis, School of Computing and Mathematics, University Plymouth, United Kingdom

149. Weifeng, Lv; Jianchu, Kang; Wei, Chen; Ran, Lei (2007), “Integration and Application Platform of Service-Oriented Telecom Businesses”, Fourth European Conference on Universal Multiservice Networks (ECUMN 2007), pp. 183-189, IEEE

150. Woyczechowski, Hilmar (2008), “Using all senses”, Detecon International GmbH, Available at: http://www.detecon-dmr.com/en/article/using-all-senses_2008_03_31, [accessed 5th November 2013]

151. Wu, Jun (2009), “Research on SOA-based service integration architecture in telecom industry”, International Conference on Service Operations, Logistics and Informatics (SOLI 2009), pp.604-608, IEEE

152. Zave, Pamela (2004), “FAQ Sheet on Feature Interaction”, Available at: http://www2.research.att.com/~pamela/faq.html, [accessed 25th June 2013], AT&T

153. Zhuang, W.; Tang, Y.; Hu, Y. (2013), “Design and implementation of SIP B2BUA server”, International Conference on Anti-Counterfeiting, Security and Identification, pp.1-5, IEEE

Appendix A – Abbreviations

3GPP	Third Generation Partnership Project

A

AAA	Authentication, Authorisation, and Accounting
ABNF	Augmented Backus-Naur Form
ACE	Automatic Composition Engine
ALG	Application Layer Gateway
ANI	Application Network Interface
API	Application Programming Interface
AS	Application Server

B

B2BUA	Back-to-Back User Agent
BIND	Berkeley Internet Name Domain
BS	Base Station
BSS	Business Support System

C

CB	Call Blocking
CFB	Call Forwarding on Busy
CIM	Context Information Management
CN	Core Network
CS	Call Server
CSCF	Call Session Control Function
CLIP	Calling Line Identification Presentation

D

DFC	Distributed Feature Composition
DNS	Domain Name System
DHT	Distributed Hash Table

E

ETSI	European Telecommunications Standards Institute

F

FE	Functional Entity

G

GSM	Global System for Mobile communications
GPRS	General Packet Radio Service

H

HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol

I

IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
ID	Identity
IMS	IP Multimedia Subsystem
IP	Internet Protocol
ISDN	Integrated Services Digital Network
IST-SPICE	Information Society Technologies – Service Platform for Innovative Communication Environment
IT	Information Technology
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector

J

JAIN	Java APIs for Integrated Networks
JAXB	Java Architecture for XML Binding
JMS	Java Message Service
JSDL	JSON Service Description Language
JSON	JavaScript Object Notation

K

L

M

MAC	Medium Access Control
MCU	Multipoint Control Unit
Megaco	Media Gateway Control Protocol
MGC	Media Gateway Controller
MGW	Media Gateway
MIME	Multipurpose Internet Mail Extensions
MMCCS	Multi-Modal Communication and Collaboration Services
MMS	Multimedia Messaging Service
MPEG	Moving Pictures Experts Group
MS	Media Server
MSC	Message Sequence Chart
MSF	MultiService Forum

N

NAL	Network Abstraction Layer
NAPT	Network Address and Port Translation
NGN	Next Generation Networks
NGSON	Next Generation Service Overlay Networks
NNI	Network Network Interface

O

OASIS	Organization for the Advancement of Structured Information Standards
OMA	Open Mobile Alliance
OSA	Open Service Access
OSA-SCS	Open Service Access – Service Capability Server
OSE	Open Service Environment
OSI	Open Systems Interconnection
OSS	Operational Support Systems
OWL-S	Ontology Web Language for Web Services

P

P2P	Peer-to-Peer
PS	Packet Switched
PSTN	Public Switched Telephone Network

Q

QoS	Quality of Service

R

RELOAD	Resource Location And Discovery
REST	Representation State Transfer
RFC	Request For Comments
RSDL	RESTful Service Description Language
RTP	Real-time Transport Protocol

S

SAP	Service Access Point
SBC	Session Border Controller
SC	Service Composition
SCE	Service Creation Environment
SCIM	Service Capability Interaction Manager

S-CSCF	Serving – Call Session Control Function
SCS	Service Capability Server
SCTP	Stream Control Transmission Protocol
SDL	Service Description Language
SDN	Service Discovery and Negotiation
SDP	Session Description Protocol
SEL	Service Exposure Layer
SGW	Signalling Gateway
SIP	Session Initiation Protocol
SIPS	Session Initiation Protocol Security
SLEE	Service Logic Execution Environment
SMS	Short Message Service
SNI	Service Network Interface
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SPATEL	SPice Advanced service description language for TELecommunication services
SPICE	Service Platform for Innovative Communication Environment
SPOF	Single Point of Failure
SR	Service Routing
SS7	Signalling System 7

T

TCP	Transmission Control Protocol
TISPAN	Telecommunications & Internet converged Services and Protocols for Advanced Networking
TLS	Transport Layer Security
TTS	Text-To-Speech

U

UAC	User Agent Client
UDDI	Universal Description, Discovery and Integration

UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
UNI	User-Network Interface
UNSPSC	United Nations Standard Products and Service Code
URI	Uniform Resource Identifier

V

VAS	Value-Added Service
VoIP	Voice over IP

W

WADL	Web Application Description Language
WG	Working Group
WSDL	Web Service Description Language

X

XML	Extensible Markup Language

Y

Z

Appendix B – Algorithm Test Result

Table B.0.1: Resolved compositions

0	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
1	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
2	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
3	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
4	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de)

	(sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
5	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
6	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
7	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
8	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
9	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de)

	(sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
10	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
11	(sip:information.greatwestern@p2p.de|sip:recogniser.att@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.alice@p2p.de|sip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
12	(sip:recogniser.bt@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
13	(sip:recogniser.bt@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
14	(sip:recogniser.bt@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
15	(sip:recogniser.bt@p2p.de|sip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.de|sip:tts.1und1@p2p.de|sip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)

16	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
17	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
18	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
19	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
20	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
21	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
22	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
23	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
24	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de)

	(sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
25	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
26	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
27	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
28	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
29	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
30	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
31	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
32	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de)

	(sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
33	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
34	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
35	(sip:recogniser.bt@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
36	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
37	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
38	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
39	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
40	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de)

	(sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
41	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
42	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
43	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
44	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
45	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
46	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
47	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
48	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de)

	(sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
49	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
50	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
51	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
52	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
53	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information.greatwestern@p2p.de)
54	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
55	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
56	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de)

	(sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
57	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
58	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
59	(sip:recogniser.telefonica@p2p.delsip:translation-en-de.leo@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information.greatwestern@p2p.de)
60	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
61	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
62	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
63	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
64	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de)

	(sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
65	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
66	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
67	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
68	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.1und1@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.1und1@p2p.de) (sip:information2.greatwestern@p2p.de)
69	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.bt@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
70	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.eplus@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)
71	(sip:information2.greatwestern@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:translation-en-de.leo@p2p.delsip:tts.alice@p2p.delsip:record.google@p2p.de) (sip:vote.o2@p2p.de) (sip:tts.alice@p2p.de) (sip:information2.greatwestern@p2p.de)

Appendix C – Service Request Results

Table C.0.1: Simple Composite Service Requests

Composition Request	Responses
(conference|information)	• sip:video.grundig@p2p.de;sip:karaoke.youtube@p2p.de • sip:video.grundig@p2p.de;sip:newsticker.eintracht@p2p.de • sip:audio.orange@p2p.de
(conference|playback)	• sip:audio.orange@p2p.de • sip:video.grundig@p2p.de
(conference|text2speech)	• sip:audio.orange@p2p.de • sip:video.grundig@p2p.de
(conference|conference)	• sip:audio.orange@p2p.de;sip:audio.orange@p2p.de • sip:video.grundig@p2p.de;sip:video.grundig@p2p.de
(information|conference)	• sip:karaoke.youtube@p2p.de;sip:video.grundig@p2p.de • sip:newsticker.eintracht@p2p.de;sip:video.grundig@p2p.de
(information|playback)	• sip:karaoke.youtube@p2p.de • sip:newsticker.eintracht@p2p.de
(information|text2speech)	• sip:newsticker.eintracht@p2p.de;sip:tts.oxford@p2p.de • sip:karaoke.youtube@p2p.de
(information|information)	• sip:karaoke.youtube@p2p.de;sip:karaoke.youtube@p2p.de • sip:karaoke.youtube@p2p.de;sip:newsticker.eintracht@p2p.de • sip:newsticker.eintracht@p2p.de;sip:karaoke.youtube@p2p.de • sip:newsticker.eintracht@p2p.de;sip:newsticker.eintracht@p2p.de
(playback|information)	• sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de • sip:music.mtv@p2p.de;sip:newsticker.eintracht@p2p.de
(playback|conference)	• sip:music.mtv@p2p.de;sip:video.grundig@p2p.de
(playback|text2speech)	• sip:music.mtv@p2p.de
(playback|playback)	• sip:music.mtv@p2p.de
(text2speech|information)	• sip:tts.oxford@p2p.de
(text2speech|conference)	• sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de
(text2speech|playback)	• sip:tts.oxford@p2p.de
(text2speech|text2speech)	• sip:tts.oxford@p2p.de

Table C.0.2: Complex Composite Service Requests

Composition Request	Responses
(conferenceInformation) (conference)	• (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de) (sip:audio.orange@p2p.de) • (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de) (sip:video.grundig@p2p.de) • (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de) (sip:audio.orange@p2p.de) • (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de) (sip:video.grundig@p2p.de) • (sip:audio.orange@p2p.de) (sip:audio.orange@p2p.de) • (sip:audio.orange@p2p.de) (sip:video.grundig@p2p.de)
(conferenceInformation) (text2speech|conference)	• (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de) (sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de) • (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de) (sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de) • (sip:audio.orange@p2p.de) (sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de)
(playback|information.name=i1) (i1|text2speech|conference)	• (sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de) (sip:karaoke.youtube@p2p.de) • (sip:music.mtv@p2p.de; sip:newsticker.eintracht@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de)
(conference.name=c1) (playback|information.name=i1) (i1|text2speech|c1)	• (sip:audio.orange@p2p.de) (sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de) (sip:karaoke.youtube@p2p.de) • (sip:audio.orange@p2p.de) (sip:music.mtv@p2p.de; sip:newsticker.eintracht@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:tts.oxford@p2p.de;sip:audio.orange@p2p.de) • (sip:video.grundig@p2p.de) (sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de) (sip:karaoke.youtube@p2p.de)

(playback|information|conference) (information.name=i1|conference) (i1|text2speech)	• (sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:karaoke.youtube@p2p.de) • (sip:music.mtv@p2p.de;sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:tts.oxford@p2p.de) • (sip:music.mtv@p2p.de; sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:karaoke.youtube@p2p.de) • (sip:music.mtv@p2p.de; sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:newsticker.eintracht@p2p.de; sip:tts.oxford@p2p.de)
conference.name=c1|information|c1	• sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de • sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de • sip:audio.orange@p2p.de
(conference.name=c1|information|c1) (c1|information)	• (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de) • (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de; sip:video.grundig@p2p.de) (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de) • (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:video.grundig@p2p.de; sip:karaoke.youtube@p2p.de)

	• (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de; sip:video.grundig@p2p.de) (sip:video.grundig@p2p.de; sip:newsticker.eintracht@p2p.de) • (sip:audio.orange@p2p.de) (sip:audio.orange@p2p.de)

Appendix D – Publications and Presentations

The following list includes publications and presentations related to the area of this research, to which the author of this thesis has contributed during the course of research.

1. Lehmann, Armin; Trick, Ulrich (2007), "Abschlussbericht Detecon – Services in NGN" (translated title: "Final Report Detecon – Services in NGN"), University of Applied Sciences Frankfurt am Main and Detecon International GmbH

2. Lehmann, Armin; Trick, Ulrich; Oehler, Steffen (2007), "Optimierung der Diensteentwicklung und –bereitstellung in UMTS/IMS-Mobilfunknetzen" (translated title: "Optimisation for service development and provisioning in UMTS/IMS mobile phone networks"), 12. VDE/ITG (Verband der Elektrotechnik Elektronik Informationstechnik/Informationstechnische Gesellschaft) Mobilfunktagung Osnabrück, Germany, ISBN: 978-3-8007-3036-0

3. Lehmann, Armin; Eichelmann, Thomas; Trick, Ulrich (2008a), "Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation" (translated title: "New possibilities for service provisioning based on Peer-to-Peer communication"), 13. VDE/ITG (Verband der Elektrotechnik Elektronik Informationstechnik/Informationstechnische Gesellschaft) Mobilfunktagung Osnabrück, Germany, ISBN: 978-3-8007-3104-6

4. Lehmann, Armin; Fuhrmann, Woldemar; Trick, Ulrich; Ghita, Bogdan (2008b), "New possibilities for the provision of value-added services in SIP-based peer-to-peer networks", Proceedings of SEIN (Symposium on Security, E-learning, Internet and Networking), University of Wrexham, United Kingdom, ISBN: 978-1-84102-196-6

5. Lehmann, Armin; Trick, Ulrich; Fuhrmann, Woldemar (2009a), "SOA-basierte Peer-to-Peer-Mehrwertdienstebereitstellung" (translated title: "SOA based peer-to-peer provisioning of value-added services"), 14. VDE/ITG (Verband der Elektrotechnik Elektronik Informationstechnik /Informationstechnische Gesellschaft) Mobilfunktagung Osnabrück, Germany, ISBN: 978-3-8007-3164-0

6. Lehmann, Armin; Fuhrmann, Woldemar; Trick, Ulrich; Ghita, Bogdan (2009b), "A new approach to classify and describe telecommunication services", Proceedings of SEIN (Symposium on Security, E-learning, Internet and Networking), University of Applied Sciences Darmstadt, Germany, ISBN: 978-1-84102-236-9

7. Lehmann, Armin; Eichelmann, Thomas; Trick, Ulrich; Lasch, Rolf; Ricks, Björn; Toenjes, Ralf (2009), "TeamCom: A Service Creation Platform for Next Generation Networks", Fourth International Conference on Internet and Web Applications and Services (ICIW), Venice, Italy, pp. 12-17, IEEE, ISBN: 978-1-4244-3851-8

8. Lehmann, Armin; Trick, Ulrich; Fuhrmann, Woldemar; Ghita, Bogdan (2011), "A new service description language as basis for service composition in SIP-based peer-to-peer infrastructures", 4th Wireless and Mobile Networking Conference (WMNC), Toulouse, France, pp. 1-8, IEEE, ISBN: 978-1-4577-1192-3

Copies of the papers most closely related to the research described are enclosed within this appendix.

Published in *ITG-Fachbericht Mobilfunk (Mobilfunktagung 2008)*, pp. 87-92, University of Applied Sciences, Osnabrück, Germany, ISBN: 978-3-8007-3104-6

Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation

Armin Lehmann, Thomas Eichelmann, Ulrich Trick
Fachhochschule Frankfurt/M. - University of Applied Sciences, Kleiststraße 3, 60318 Frankfurt/M., Germany
E-Mail: lehmann@e-technik.org, eichelmann@e-technik.org, trick@e-technik.org

Das dieser Publikation zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 1704B07 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Kurzfassung

Die Bereitstellung neuer Telekommunikationsdienste, speziell Mehrwertdienste, wird in Zukunft eine entscheidende Rolle spielen. Hierüber werden sich Provider von ihren Konkurrenten abgrenzen. Durch Erkenntnisse aus Peer-to-Peer-Kommunikationsnetzen wird ein neuer und einfacher Weg zur Dienstebereitstellung aufgezeigt. Speziell für das IP Multimedia Subsystem (IMS) wird anhand eines Beispiels dargestellt, wie auch hier eine Umsetzung der einfachen Bereitstellung realisiert werden kann.

1 Einleitung

Die Bereitstellung von Diensten, insbesondere von Mehrwertdiensten (Value-added Services (VAS)) in zukünftigen Netzen, speziell in IP-basierten UMTS-Mobilfunknetzen, wird eine entscheidende Rolle spielen. Ein wichtiges Unterscheidungsmerkmal bei den Providern wird zukünftig das Angebot an Diensten sein.

Dieses Angebot wird durch die steigende und spezialisierte Nachfrage der Kunden bestimmt. Die Kunden werden in Zukunft Dienste fordern, die für sie maßgeschneidert wurden. Um diesen wachsenden Anforderungen zu genügen, ist eine besonders wichtige Anforderung an zukünftige Netze ihre „Offenheit für neue Dienste“.

Hieraus ergibt sich eine ganze Reihe von Fragen. Wie können neue Dienste auf eine einfache Art und Weise bereitgestellt werden? Welche Voraussetzungen müssen die Provider erfüllen, um diese Art der Bereitstellung von neuen Diensten zu gewährleisten? Welche Möglichkeiten bietet hierbei das IP Multimedia Subsystem (IMS) und welche Chancen ergeben sich durch die Anwendung der Peer-to-Peer-Kommunikation? Diese Fragestellungen und Antworten darauf werden im Folgenden erörtert.

Durch die zukünftigen Netze und deren Diensteplattformen (Service Delivery Platform, SDP) wird eine Bereitstellung von sogenannten Mehrwertdiensten erleichtert. Mehrwertdienste sind spezielle Telekommunikationsdienste, die weit über die Funktionalität reiner Basisdienste, wie z.B. Telefon- oder Telefaxdienst, hinausgehen. Damit VAS bereitgestellt werden können, werden Funktionen benötigt, die das Kernnetz alleine nicht bereitstellen kann. Hierfür werden, wie in Bild 1 dargestellt, weitere Netzelemente wie Application Server (AS) und Media Server (MS) benötigt.

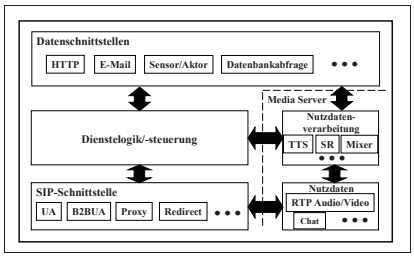


Bild 1 Application (AS) und Media Server (MS)

Die AS stellen die Dienste zur Verfügung und bedienen sich bei Bedarf für komplexere Dienste weiterer AS oder MS. Der Media Server stellt hierbei die Funktionalität zur Verarbeitung der Nutzdaten z.B. bei Video-Konferenzen bereit.

2 Übersicht zu verschiedenen Peer-to-Peer-Modellen

Auf Grund der Vorteile wie z.B. Skalierbarkeit, geringe Kosten und Offenheit für neue Dienste werden Peer-to-Peer- (P2P) Modelle zukünftig eine wichtige Stellung einnehmen. Daher werden in diesem Abschnitt verschiedene Peer-to-Peer-Konzepte diskutiert und auf ihre Möglichkeiten zur Dienstebereitstellung hin untersucht.

Zunächst können Peer-to-Peer-Architekturen in die folgenden drei Klassen (hier: Generationen) unterteilt werden, wie in Bild 2 zu sehen ist [1].

1st Generation (Hybrid P2P-Modell):
- z.B. Napster, Skype, SIP
- Client-Server- und Peer-to-Peer-Kommunikation
- semi-zentral (mindestens ein zentraler Kontrollpunkt, z.B. Indexserver)

2nd Generation (Pure P2P-Modell):
- z.B. Gnutella
- völlig dezentral
- Organisation der Peers kann strukturiert oder unstrukturiert sein

3rd Generation (Super P2P-Modell):
- z.B. KaZaa
- Weiterentwicklung des Hybrid P2P-Modells zu einem hierarchischen Modell
- Zentrale Kontrollpunkte werden mit einem P2P-Netzwerk aus sogenannten Super-Nodes getauscht
- Super-Nodes und normale Peers interagieren in Client-Server-Beziehung

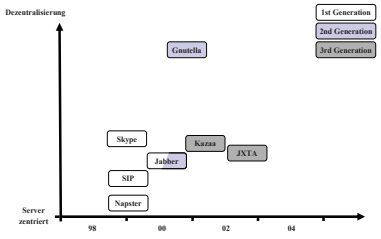


Bild 2 Peer-to-Peer-Generationen

Die verschiedenen Peer-to-Peer-Varianten sind zur besseren Übersicht nochmals in Bild 3 dargestellt.

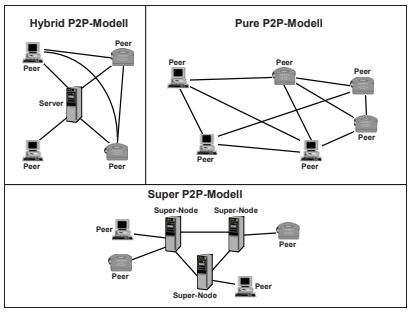


Bild 3 Übersicht der Peer-to-Peer-Varianten

Bezüglich der Anforderungen aus Sicht eines öffentlichen Kommunikationsnetzes wurden die verschiedenen P2P-Ansätze im Rahmen des BMBF-Projekts "IMS- oder P2P-basierte Dienstebereitstellung und -entwicklung für kundenspezifische Kommunikationsprozesse (TeamCom)" analysiert. Hierbei stellte sich heraus, dass aus Netzwerksicht das Hybrid P2P-Modell favorisiert wird (siehe Tabelle 1).

Tabelle 1 Anforderungen an Netze

Anforderungen	Hybrid P2P	Super P2P	Pure P2P
Offenheit für neue Dienste	++	++	++
Basisdienste	++	++	++
Telefondienstmerkmale	++	++	+
Mehrwertdienste	++	++	+
Mobilität	++	+	0
Regulatorische Anforderungen	+	-	--
Integrierte Sicherheitsfunktionen	++	+	-
Den Diensten angemessene Entgelterfassung	++	+	0

Die dargestellte Tabelle ist nur ein kleiner Ausschnitt aus der im Projekt erarbeiteten Anforderungsliste. Dieser Ausschnitt zeigt allerdings schon die Vorzüge des Hybrid-Modells. Wichtige Kriterien wie z.B. Mobilität (hierzu zählen Dienstemobilität, Session-Mobilität und Persönliche Mobilität), die Erfüllung regulatorischer Anforderungen (z.B. Lawful Interception), Integrierte Sicherheitsfunktionen (z.B. Authentifizierung oder Verschlüsselung von Nutzdaten und Signalisierung) und den Diensten angemessene Entgelterfassung werden durch das hybride P2P-Modell besser unterstützt als durch die dezentraleren Varianten.

Wie stellt sich nun die Architektur eines solchen Hybrid-Modells bezüglich der Dienstebereitstellung dar. Dieser Aspekt wird im folgenden Abschnitt eingehend erläutert.

3 Dienstebereitstellung in einem Hybrid P2P-Modell

Basierend auf dem heutigen SIP-Standard [2] kann ein Hybrid P2P-Modell aufgebaut werden. SIP unterstützt bereits weitestgehend P2P-Kommunikation (siehe Bild 3), da im Standard kein klares Client-Server-Modell, sondern als Normalfall ein Hybrid P2P-Modell spezifiziert wurde (siehe Bild 4). Aufbauend auf den vier Netzelementen SIP User Agent, SIP Proxy Server, SIP Registrar Server und Location Server kann eine P2P-Infrastruktur aufgebaut werden. Diese Infrastruktur ermöglicht bereits eine Client-to-Client-Kommunikation (siehe Bild 4).

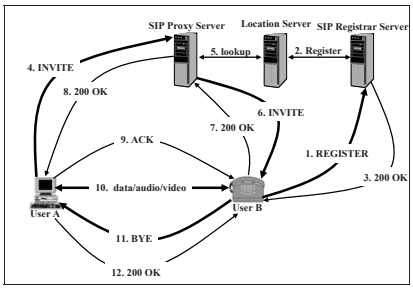


Bild 4 SIP-basierte Client-to-Client-Kommunikation

Wie in Bild 4 zu erkennen, dienen der SIP Registrar und SIP Proxy Server ausschließlich zur Ermittlung der temporären SIP URI (Uniform Resource Identifier) des Angerufenen (SIP User Agent B) und des Routens der initialen SIP-Nachrichten (hier: INVITE und 200 OK). Deutlich zu erkennen ist, dass bereits der Abschluss des Three-Way-Handshakes (mit ACK) direkt zwischen den einzelnen Clients bzw. Peers vonstatten geht. Ab diesem Zeitpunkt werden alle Nutzdaten und Signalisierungen zwischen den Clients Peer-to-Peer ausgetauscht.

Wie kann nun eine solche Hybrid P2P-Infrastruktur eingesetzt werden, um möglichst einfach und schnell Dienste, speziell Mehrwertdienste bereitstellen zu können?

Zunächst wird ein P2P-Overlay bestehend aus mehreren SIP-Registrar/Proxy Servern erstellt. Diese sind alle mittels DNS (Domain Name System) [3; 4] oder Dynamic DNS [5] unter einer Domain zu finden. Somit können sich SIP-Netzelemente (z.B. SIP User Agents) einfach an diesem Overlay-Netz registrieren. Angebunden an das Overlay ist ein Location Server-Pool, in dem die Registrierungsdaten für das Routing abgelegt werden. Durch eine einfache Anbindung von Application Server und Media Server als Peers können Mehrwertdienste bereitgestellt werden. Diese Architektur ist in Bild 5 dargestellt.

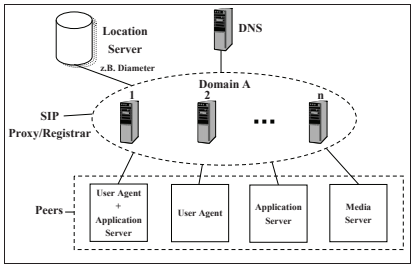


Bild 5 Hybrid-P2P-Modell

In Bild 6 ist die prinzipielle Architektur eines NGN (Next Generation Networks) dargestellt (vgl. Bild 8). Das in diesem Abschnitt beschriebene Hybrid-P2P-Modell kann hierbei eine spezielle Ausprägung des NGN repräsentieren.

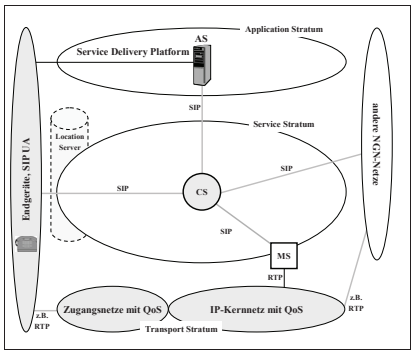


Bild 6 NGN-Architektur in einer Strata-Struktur

Damit nun aber auch die Application Server und Media Server anhand von SIP URIs gefunden werden können, müssen auch diese sich am Overlay-Netz registrieren (siehe Bild 5).
Diese Architektur bietet die nachfolgend genannten Vorteile:

- Verwendung von bestehenden Standards wie SIP und DNS

- Basierend auf dem SIP-Standard können nicht-modifizierte SIP User Agents genutzt werden.
- Lookups (Auflösung der ständigen SIP URI in die temporäre URI) gehen schneller vonstatten als in reinen P2P-Modellen.
- Engpässe und Single Points of Failure werden eliminiert.
- Komplexere Suchen im Netz sind möglich.
- Dienste können registriert und deregistriert werden.
- Kostenteilung/ -reduktion
- Skalierbarkeit

4 Neue Möglichkeiten der Dienstebereitstellung in P2P- und IMS-Netzen

Damit die neuen Dienste, die auf den SIP Application Servern implementiert wurden, für die Endnutzer erreichbar werden, müssen die AS um eine weitere Funktionalität erweitert werden. Der entsprechende AS sendet hierzu die Nachricht REGISTER, um sich am Netz zu registrieren (z.B. SIP Digest [2]). Um jetzt die implementierten Mehrwertdienste am Netz anzumelden, wird dieselbe Nachricht verwendet. Jedoch ist hierbei eine URI zur Registrierung bzw. Authentifizierung zu verwenden, die angelehnt ist an dem Modell der Subdomains (z.B. "sub.domain.de"). Dies wird hier an einem Beispiel verdeutlicht.

Angenommen der entsprechende Application Server ist mit der URI "as1@DomainA" am Hybrid P2P-Netz registriert. Nun sendet er eine SIP-Nachricht REGISTER an den vorab per DNS ermittelten Registrar Server mit folgendem beispielhaftem Inhalt:

REGISTER sip:DomainA SIP/2.0
To: <sip:dienstX.as1@DomainA>;tag=1234
Contact: <sip:dienstX.as1@88.88.88.88>
Expires: 3600

Mittels dieses Modells (siehe Bild 7) können nun die angemeldeten AS beliebig viele Dienste am Netz registrieren. Anhand des SIP-Header-Felds "Expires:" wird der Zeitraum für die Gültigkeitsdauer der Registrierung angegeben. In diesem Fall beträgt die Dauer 3600 Sekunden (= 1 Stunde). Hieraus ergibt sich, dass der Application Server in regelmäßigen Abständen den Dienst neu registrieren muss, solange der Dienst erreichbar sein soll. Auf genauso einfache Weise kann der AS auch wieder einen Dienst vom Netz abmelden, indem er in das Header-Feld "Expires:" eine "0" einträgt.

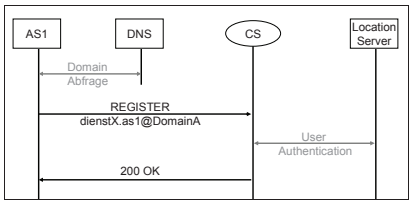


Bild 7 Diensteregistrierung im Hybrid-P2P-Modell

In dem also Application Server die auf ihnen realisierten Dienste in einem SIP-Netz mittels SIP URI registrieren, können diese Dienste auf einfachste Art und Weise in das Netz eingebracht und von SIP User Agents, aber auch anderen Application Servern genutzt werden. Dieser Mechanismus unterstützt in hervorragender Weise die „Offenheit für neue Dienste".

Wie aber kann dieser Mechanismus in ein IMS übernommen werden? Hierzu wird die Infrastruktur eines IMS, welches dem SIP-Standard angelehnte logische Netzelemente beherbergt, im Folgenden näher erläutert. In Bild 8 ist die prinzipielle Architektur eines NGN (Next Generation Networks) mit IMS dargestellt [6].

P-CSCF (Proxy-Call Session Control Function):
- Arbeitet normalerweise als SIP Proxy Server und leitet Nachrichten an entsprechende I- bzw. S-CSCFs weiter
- Erste Anlaufstelle für UE (User Equipment, UMTS-Endgerät)
- Zuständig für Accounting-Funktionen

I-CSCF (Interrogating-CSCF)
- Arbeitet als SIP Proxy Server und leitet Nachrichten an S-CSCFs weiter
- Dient als Schnittstelle zu anderen IMS bzw. IP Multimedia-Netzen

S-CSCF (Serving-CSCF):
- Agiert als SIP Proxy/Registrar Server und Back-to-Back User Agent
- Steuert SIP-Verbindungen/ Dienste/ Dienstmerkmale, kommuniziert dazu mit UE, anderen CSCFs und AS
- Nutzerdaten bzw. Filterkriterien zur Auswahl der AS werden bei Registrierung vom HSS (Home Subscriber Server) in die S-CSCF geladen. Diese AS-Filterkriterien sind Teil des Service-Profils eines Nutzers.
- Zuständig für Accounting-Funktionen
- Kommuniziert mit HSS per Diameter-Protokoll

HSS (Home Subscriber Server):
- Zentrale Datenbank (vgl. Location Server)

- Enthält Nutzeridentitäten, Registrierungsinformationen, Zugriffsrechte, Service Trigger-Informationen für IMS
- Zugriff durch z.B. CSCFs, AS
- Wird mittels Diameter-Protokoll angesprochen
- Authentifizierung für alle Strata über HSS

MRF (Media Resource Function):

- Media Server (MS)
- Für Sprachaufzeichnung und -wiedergabe, Videoaufzeichnung und -wiedergabe, Spracherkennung, Konvertierung von Text in Sprache, Multimedia-Konferenzen, Transcoding multimedialer Daten
- MRF ist aus S-CSCF und AS-Sicht Slave
- Wird mittels SIP durch S-CSCF bzw. AS gesteuert

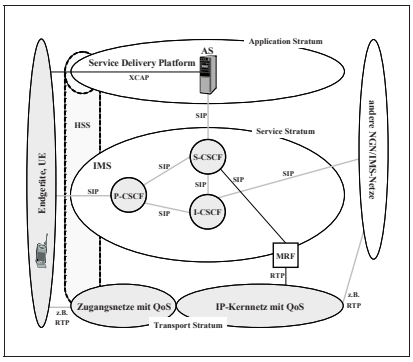


Bild 8 Architektur eines NGN mit IMS

Ähnlich der Methode wie beim Hybrid-P2P-Modell können neue Dienste auch an einem IMS registriert werden. Dies wird hier ebenfalls exemplarisch dargestellt.

Angenommen ein AS, der nicht zum IMS des Providers zählt, hat sich bereits mittels der SIP-Nachricht REGISTER mit der URI "as_extern1@ims.de" registriert. Im weiteren Verlauf sendet dieser AS eine weitere SIP-Nachricht REGISTER an das IMS, um den neuen Dienst "dienstX" zu registrieren (siehe Bild 9).

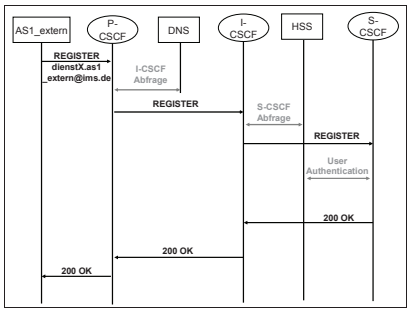


Bild 9 Diensteregistrierung im IMS

Diesem Beispiel ist zu entnehmen, dass mittels exakt derselben Methode, wie im Hybrid-Modell erläutert, auf eine sehr einfache Art und Weise neue Dienst auch an einem IMS registriert werden können.

Auch wichtige Sicherheitsaspekte können weiterhin einfach umgesetzt werden. So kann z.B. der Betreiber des IMS verlangen, das eine Authentifizierung bei der Anmeldung bzw. beim Abruf eines Dienstes stattfindet (SIP Digest [2]).

Für das Abrechnen der Nutzung der Dienste können verschiedene Szenarien in Frage kommen. Z.B. kann der Provider des IMS dem Betreiber des externen AS anbieten, für diesen die Abrechnung zu tätigen, da der IMS-Provider über die dafür nötigen Informationen, z.B. gespeicherte Dienstabrufe, verfügt. Der externe Diensteanbieter kann aber auch eigenständig Rechnungen für die Nutzung seiner Dienste an die Endkunden richten, indem er verlangt, dass sich die Endkunden vorab bei ihm mit den notwendigen Daten registrieren müssen.

Durch Nutzung dieser Ergebnisse kann ein Provider seinen Kunden oder anderen Providern ganz einfach neue Mehrwertdienste anbieten und bereitstellen. Insbesondere kann hierdurch die Offenheit für neue Dienste in IMS-Netzen verbessert werden.

5 Literatur

[1] Eisenschmid, Wilhelm: Peer-to-Peer-Architekturen. Universität Ulm, Proseminar Virtuelle Präsenz, 2005

[2] Rosenberg, J.; Schulzrinne, H.; Camarillo, G.; Johnston, A.; Peterson, J.; Sparks, R.; Handley, M.; Schooler, E.: RFC 3261 - SIP: Session Initiation Protocol. IETF, June 2002

[3] Mockapetris, P.: RFC 1034 - DOMAIN NAMES - CONCEPTS AND FACILITIES. IETF, November 1987

[4] Mockapetris, P.: RFC 1035 - DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION. IETF, November 1987

[5] Vixie, P.; Thomson, S.; Rekhter, Y.; Bound, J.: RFC 2136 - Dynamic Updates in the Domain Name System (DNS UPDATE). IETF, April 1997

[6] Trick, Ulrich; Weber, Frank: SIP, TCP/IP und Telekommunikationsnetze. Oldenbourg, April 2007

6 Abkürzungen

AS	Application Server
B2BUA	Back-to-back User Agent
BMBF	Bundesministerium für Bildung und Forschung
CSCF	Call Session Control Function
DNS	Domain Name System
HSS	Home Subscriber Server
I-CSCF	Interrogating-Call Session Control Function
IMS	IP Multimedia Subsystem
IP	Internet Protocol
MRF	Media Resource Function
MS	Media Server
NGN	Next Generation Networks
P2P	Peer-to-Peer
P-CSCF	Proxy-Call Session Control Function
RTP	Real-Time Transport Protocol
S-CSCF	Serving-Call Session Control Function
SDP	Service Delivery Platform
SIP	Session Initiation Protocol
SR	Speech Recognition
TTS	Text-to-Speech
UA	User Agent
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
VAS	Value-added Service

Published in *Proceedings of the Fourth Collaborative Research Symposium on Security, E-learning, Internet and Networking (SEIN 2008)*, pp. 166-175, Glyndŵr University, Wrexham, UK, ISBN: 978-1-84102-196-6

New possibilities for the provision of value-added services in SIP-based peer-to-peer networks

A.Lehmann[1,2], W.Fuhrmann[3], U.Trick[1], B.Ghita[2]

[1] Research Group for Telecommunication Networks, University of Applied Sciences Frankfurt/M., Frankfurt/M., Germany
[2] Network Research Group, University of Plymouth, Plymouth, United Kingdom
[3] University of Applied Sciences Darmstadt, Darmstadt, Germany
e-mail: lehmann@e-technik.org

Abstract

Service provisioning in future telecommunication networks, especially value-added services, is becoming more and more important. As a result providers will have to distinguish themselves sharply from their competitors. Newest insights based on the research of peer-to-peer networks will present an easy way to service provisioning. Furthermore this paper will present a solution for service discovery and the reuse of still existing services. These solutions are still the basis for future work which will have the goal to create a framework which will provide also the composition of new services in peer-to-peer networks.

Keywords

NGN, SIP, Value-added services, peer-to-peer networks

1. Introduction

Service provisioning, especially value-added services (VAS), will play a key role in Next Generation Networks (NGN). An important distinguishing feature for providers will be the range of new services. This offer is determined by the rising and specified inquiry of the customers. In future customers will demand services which became tailor-made for them. To meet these increasing requirements, one of the critical steps is to implement in the future networks the support for openness towards new services. This will lead to a number of challenges, such as the management of new services, the conditions to be fulfilled by providers to ensure their provisioning, and the impact of involving peer-to-peer (P2P) technologies in such architectures. The aim of this paper is to review and discuss all these challenges
Through future networks and their service delivery platforms (SDP) the provision of value-added services will be facilitated. Value-added services are specialized telecommunications services. Their functionality will exceed teleservices, e.g. facsimile or short message service. To provide VAS there is a need for new functions which are not handled by the core network itself. For this purpose further network elements such as application servers (AS) and media servers (MS) are needed as shown in Fig.1.

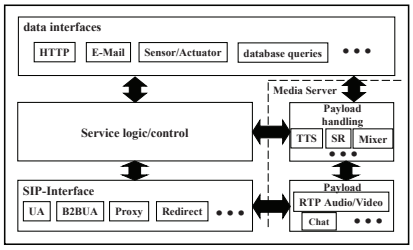


Figure 1: Application and Media Server

The application servers will provide the services and will use/control other application servers and/or media servers to realize more complex services. Media servers will provide the handling of payload (e.g. for video conferences).

2. Overview of different peer-to-peer models

Based on advantages such as scalability, low costs, and openness for new services peer-to-peer models will play an important role in the future. Thus, we will discuss different peer-to-peer models in this section and will determine their possibilities of service provisioning. Three different models (generations) were proposed by prior research (Eisenschmid, 2005) as shown in Fig.2 (E. Harjula *et al.*, 2004).

1st Generation (Hybrid P2P model):

- e.g. Napster, Skype, SIP (Session Initiation Protocol)
- semi-centric (at least one centric checkpoint), e.g. one centric peer as an index server for available data
- client-server and P2P communication
- there is no need for specialized tracing service

2nd Generation (Pure P2P model):

- e.g. Gnutella
- fully distributed
- structured (linked to a special topology) or non-structured organization of peers
- no support for searching by keywords
- transient peers are not supported

3rd Generation (Super P2P model):

- e.g. KaZaa, Skype
- enhancement of the hybrid P2P model
- centralized checkpoints are substituted by P2P network of super nodes
- interaction between super nodes and normal peers is based on client-server model

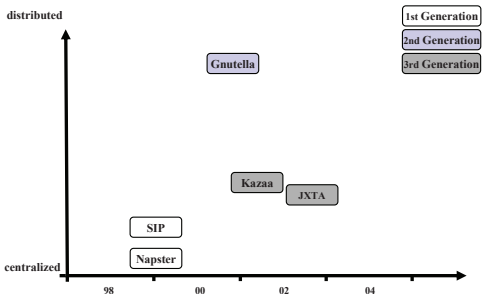


Figure 2: Peer-to-peer Generations

A summary of these different peer-to-peer variants is shown in Fig. 3.

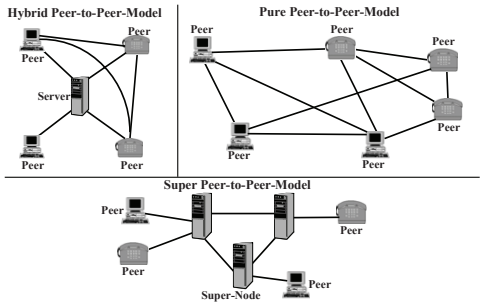


Figure 3: Overview of different Peer-to-Peer variants

With respect to the requirements from the view of public communication networks the different P2P variants have been analyzed within the research project TeamCom (Lehmann, 2008). In this connection the hybrid P2P model is favored by the networks view listed in Table 1 (Lehmann, 2008).

Requirements	Hybrid P2P	Super P2P	Pure P2P
Openness for new services	++	++	++
Teleservices	++	++	++

Supplementary services	++	++	+
Value-added services	++	++	+
Mobility	++	+	0
Regulatory Requirements	+	-	--
Integrated Security Functions	++	+	-
appropriate charging for services	++	+	0

Table 1: Requirements for Networks

The above shown table is only an extract of the requirements list developed in the project. But this extract already illustrates the preferences of the hybrid model. Major criteria e.g. mobility (service mobility, session mobility and personal mobility), compliance to regulatory requirements (such as lawful interception), integrated security functions (such as authentication or encryption of payload and signalling) and appropriate charging for services are better supported by the hybrid P2P model than by the distributed variants. Furthermore we will have a look on the architecture of such a hybrid model for service provisioning.

3. Service provisioning in Hybrid P2P model

Based on the SIP specification (Rosenberg *et al.*, 2002) a hybrid P2P model can be realized. SIP essentially supports P2P communication as shown in Fig. 3 because a clearly specified client-server model does not exist. According to (Rosenberg *et al.*, 2002) a hybrid SIP P2P model has already been defined as shown in Fig. 4. Based on the typical SIP elements (User Agent, Proxy Server, Registrar Server, and Location Server) a P2P infrastructure can be realized. This infrastructure already enables client-to-client communication as shown in Fig. 4.

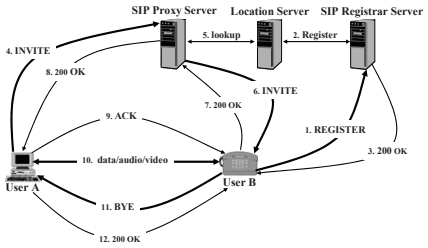


Figure 4: SIP-based client-to-client communication

As shown in Fig. 4, the SIP Registrar and SIP Proxy servers are only needed to lookup the location of the callee (SIP User Agent B). Therefore the temporary SIP URI (Uniform Resource Identifier) is resolved to route the initial SIP requests (INVITE and 200 OK). It is obvious that the three-way-handshake (by sending ACK) is completed directly between the single peers respectively clients. From this point all data (payload and signalling) will be exchanged in peer-to-peer manner.

Such a hybrid P2P infrastructure can be used to provide quick and easy services, especially value-added services.

First a P2P overlay composed of SIP Registrar/Proxy servers has to be created. The domain for these elements can be resolved by DNS (Domain Name System) (Mockapetris, 1987, I), (Mockapetris, 1987, II) or dynamic DNS (Vixie *et al.*, 1997). Now further SIP network elements can simply register with the overlay network. A pool of location servers is connected to the overlay. The location server pool is holding the data of the registrations from the peers to provide the routing. Now application servers and media servers can easily be connected to the overlay as peers to provide value-added services. An overview of this architecture is presented in Fig. 5.

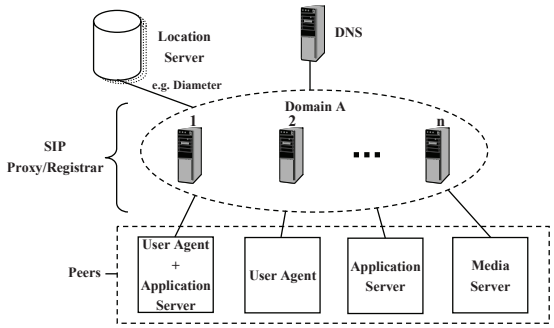


Figure 5: Hybrid SIP P2P Model

The general architecture of NGN is presented in Fig. 6. The described hybrid P2P model can represent a special characteristic of NGN.

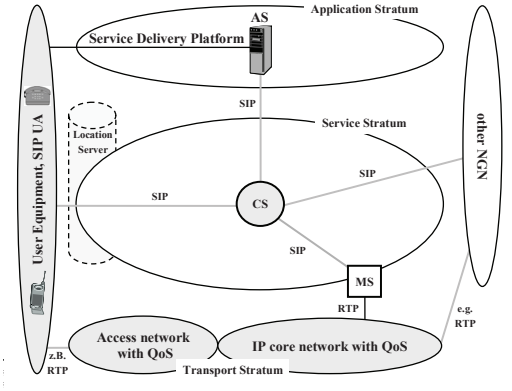


Figure 6: NGN architecture as a strata-structure

To manage the addressability of application and media servers by their SIP URI these servers also have to register with the overlay network. This architecture offers the following benefits:

- usage of existing specifications such as SIP, DNS

- non-modified standard SIP User Agents can be used (based on the SIP specification)
- fast lookups (resolving the temporary SIP URI from the permanent SIP URI) are provided better than by pure P2P models
- bottlenecks and single points of failure are eliminated
- possibility for more complex searching
- possibility to register and deregister new services
- reduction of expenses
- scalability

To offer access to the new services to end users implemented in SIP application servers, application servers must be extended by a new functionality. An application server uses the request REGISTER, such as a SIP Digest (Rosenberg *et al.*, 2002), in order to register with the network. To sign on implemented services the same request as shown above is used (Schmidt *et al.*, 2006). The URI to use for registration and authentication respectively originates from the subdomain model (e.g. "sub.domain.de"). The following example will clarify the usage. Assuming that an application server has registered the following URI "as1@DomainA", he sends the following SIP request to the SIP Proxy resolved by DNS:

REGISTER sip:DomainA SIP/2.0

To: <sip:serviceX.as1@DomainA>;tag=1234
Contact: <sip:serviceX.as1@88.88.88.88>
Expires: 3600

By using the model presented in Fig. 7 registered application servers can provide any number of services. The SIP header field "Expires:" sets the period of time for the validity of the registration which in this case is valid for one hour. As a result the application server has to register the service in defined intervals of time to set the service accessible. With the same SIP request the service can also be deregistered from the network, setting the SIP header field "Expires:" to "0".

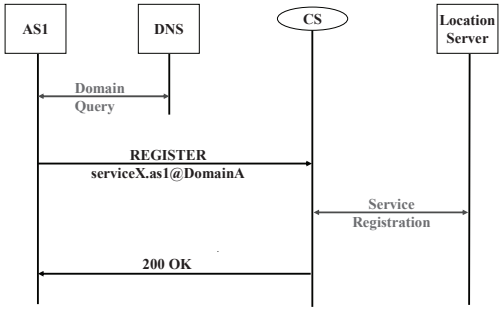


Figure 7: Service registration in hybrid P2P model

The provided services hosted by the application servers can be easily applied and SIP User Agents or other application servers can make use of them. This mechanism fully supports "openness for new services".

The next section will describe the procedures used for publishing and discovery of services.

4. Service Publishing and Discovery

Publishing and discovery of services is an essential part of a service provisioning architecture as mentioned before. To realize an architecture which supports publishing and discovery of services the overlay network has to be extended with some new features. During the past years the Internet Engineering Task Force (IETF) has standardized SIP and several extensions, particularly the SIP event framework (Roach, 2002) as an architecture for status delivery from and to any host in IP-networks and the SIP extension for event state publication (Niemi, 2004). Using these standards it is possible to create such a service provisioning architecture.

For realizing the publication of new services the SIP PUBLISH method will be used. The body of this SIP method contains the service description (such as the service name, human readable service description, service id, and service URI) specified by the use of XML (Extensible Markup Language) (Rahman *et al.*, 2006). The

PUBLISH method is sent to the overlay network. The SIP Proxy /Registrar servers are extended by a so-called SIP event server to support the publishing and discovery of services. All information belonging to the events are hosted by the SIP event server (see Fig.8).

To realize the discovery functionality two further SIP messages are used, namely SUBSCRIBE and NOTIFY. The SIP message SUBSCRIBE will be used to do a service query and the SIP message NOTIFY will be used to answer the service query. In the following the service discovery is described. A subscriber (here: SIP UA) sends the SUBSCRIBE message to initially subscribe to an event (here: service query) and receives NOTIFY for the initial notification (here: service information) and all subsequent messages that relate to this subscription. Each subscription carries a desired lifetime of the subscription. In case of a lifetime of zero, the SIP event server merely sends back all available and matching services at this point of time. If the lifetime is greater zero, an availability subscription to services in the future is established (Rahman *et al.*, 2006; Trossen and Pavel, 2005). The concluding hybrid P2P architecture is shown in Fig.8.

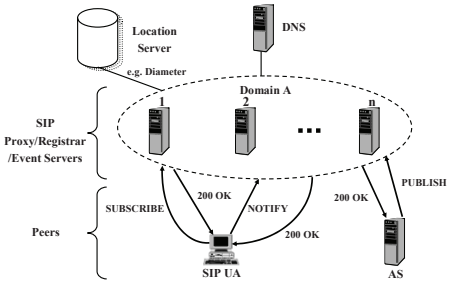


Figure 8: Hybrid P2P architecture for service provisioning and discovery

As an example the SIP messages PUBLISH, SUBSCRIBE and NOTIFY are shown in the following.

PUBLISH sip:servicex.as1@DomainA.de SIP/2.0
To: < sip:servicex.as1@DomainA.de>
From: < sip:servicex.as1@DomainA.de>;tag=1234
Expires: 3600
Event: x-servinfo
Content-Type: application/servinfo+xml
Content-Length: ...
[Published document]

SUBSCRIBE sip:servicex.as1@DomainA.de SIP/2.0
To: < sip:servicex.as1@DomainA.de>
From: <sip:userA@DomainA.de>;tag=4567

Event: x-servinfo
Accept: application/servinfo+xml
Expires: 0
Content-Length: 0

NOTIFY sip:userA@DomainA.de SIP/2.0
To: <sip:userA@DomainA.de>;tag=4567
From: < sip:servicex.as1@DomainA.de>
Event: x-servinfo
Content-Type: application/servinfo+xml
Content-Length: ...
[Published document]

Within the research project TeamCom (Lehmann, 2008) the above described architecture shown in Fig. 5 will be realized with the JAIN (Java APIs for Integrated Networks) SLEE (Service Logic and Execution Environment). This API (Application Programming Interface) uses SBBs (Service Building Block), which may represent services. These SBBs can also be shared via the presented model. Within the body of both PUBLISH and NOTIFY methods the location (e.g. an FTP (File Transfer Protocol) URI) from where the SBB can be obtained can be included, so that other peers may use these SBB and deploy it on their own application servers to also provide the same service.

5. Conclusion

With the approaches presented in this paper any service provider has the ability to easily make new value-added services accessible for users or other providers. Users can search for new services for their own use or implementation. Also the whole architecture benefits from the advantages provided by the P2P networks. Furthermore the presented architecture should be verified against still existing centralised models.

6. Annotation

The research project providing the basis for this publication was partially funded by the Federal Ministry of Education and Research (BMBF) of the Federal Republic of Germany under grant number 1704B07. The authors of this publication are in charge of its content.

7. References

Eisenschmid, W. (2005), "Peer-to-Peer-Architekturen", Universität Ulm - Ulm University, Proseminar Virtuelle Präsenz

E. Harjula et al. (2004), "Plug-and-Play Application Platform: Towards Mobile Peer-to-Peer", 3rd International conference on Mobile and Ubiquitous multimedia (MUM2004), College Park, Maryland, USA

Lehmann, A., Eichelmann, E., Trick, U. (2008), "Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation", 13. VDE/ITG Mobilfunktagung Osnabrück

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E. (2002), RFC 3261, "SIP: Session Initiation Protocol", IETF

Mockapetris, P. (1987), (I), RFC 1034, "DOMAIN NAMES - CONCEPTS AND FACILITIES", IETF

Mockapetris, P. (1987), (II), RFC 1035, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION", IETF

Vixie, P., Thomson, S., Rekhter, Y., Bound, J. (1997), RFC 2136, "Dynamic Updates in the Domain Name System (DNS UPDATE)", IETF

Schmidt, H., Guenkova-Luy, T., Hauck, F.J. (2006), "Service Location using the Session Initiation Protocol (SIP)", International conference on Networking and Services (ICNS)

Roach, A. B. (2002), RFC 3265, "Session Initiation Protocol (SIP)-Specific Event Notification", IETF

Niemi, A. (2004), RFC 3903, "Session Initiation Protocol (SIP) Extension for Event State Publication", IETF

Rahman et al. (2006), "Service Discovery using Session Initiation Protocol (SIP)", United States Patent Application Publication

Trossen, D., Pavel, D. (2005), "Service discovery & availability subscriptions using the SIP event framework", IEEE International Conference on Communications

Published in *ITG-Fachbericht Mobilfunk (Mobilfunktagung 2009)*, pp. 79-84, University of Applied Sciences, Osnabrück, Germany, ISBN: 978-3-8007-3164-0

SOA-basierte Peer-to-Peer-Mehrwertdienstebereitstellung

Armin Lehmann, Ulrich Trick
Fachhochschule Frankfurt/M. - University of Applied Sciences, Kleiststrasse 3, 60318 Frankfurt/M., Germany
Woldemar, Fuhrmann
Hochschule Darmstadt - University of Applied Sciences, Haardtring 100, 64295 Darmstadt, Germany
E-Mail: lehmann@e-technik.org, trick@e-technik.org, w.fuhrmann@fbi.h-da.de

Das dieser Publikation zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 1704B07 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren

Kurzfassung

Zukünftig wird das Bereitstellen, Finden und Kombinieren von Telekommunikationsdiensten, speziell Mehrwertdiensten, eine entscheidende Rolle spielen. Auf Basis der Möglichkeiten von Peer-to-Peer-Kommunikationsnetzen wird unter Einbeziehung der Service-orientierten Architektur (SOA) ein einfaches Verfahren für die Bereitstellung, das Finden und das Kombinieren von neuen Diensten aufgezeigt. Anhand eines Beispiels wird die Umsetzung dieser Erkenntnisse dargestellt.

1 Einleitung

Zukünftig wird das Bereitstellen, Finden und Kombinieren von Diensten, speziell Mehrwertdiensten (Value-added Services (VAS)), in SIP-basierten Netzen wie UMTS-Mobilfunknetzen ab Release 5 eine entscheidende Rolle spielen. Das Angebot an Diensten wird eines der wichtigsten Unterscheidungsmerkmale zwischen Providern werden.

Da die Next Generation Networks (NGN) Möglichkeiten bieten, Dienste auf eine einfache Art und Weise bereitzustellen und zu kombinieren, wird die Zahl der im Netz befindlichen Dienste rapide zunehmen. Auch die Nachfrage nach immer spezialisierteren Diensten wird steigen.

Hieraus ergibt sich eine ganze Reihe an Fragen. Wie werden die Nutzer neue Dienste in diesen Netzen finden können? Welche Voraussetzungen benötigen die Provider, um die Bereitstellung, das Finden und Kombinieren neuer Dienste zu gewährleisten? Welche Möglichkeiten bietet hierbei das IP Multimedia Subsystem (IMS), und welche Chancen ergeben sich in diesem Zusammenhang durch die Anwendung der Peer-to-Peer-Kommunikation? Diese Fragestellungen und Antworten darauf werden im Folgenden erörtert.

Mittels der Diensteplattformen (Service Delivery Platform, SDP) der zukünftigen Netze wird die Bereitstellung von sogenannten Mehrwertdiensten erheblich erleichtert. Mehrwertdienste sind spezielle Telekommunikationsdienste, die weit über die Funktionalität reiner Basisdienste wie z.B. Telefon- oder Telefaxdienst hinausgehen. Damit VAS bereitgestellt werden können, werden Funktionen benötigt, die das Kernnetz alleine nicht bereitstellen kann. Hierzu werden weitere Netzelemente wie Application Server (AS) und Media Server (MS) benötigt. Die AS stellen die Dienste zur Verfügung und bedienen sich bei Bedarf für komplexere Dienste weiterer AS oder MS. Der Media Server stellt hierbei die Funktionalität zur Verarbeitung der Nutzdaten z.B. bei Video-Konferenzen bereit.

2 Service-orientierte Architektur

Eine Service-orientierte Architektur ist eine Architektur, deren zentrales Konstruktionsprinzip Services (Dienste) sind. Dienste sind klar gegeneinander abgegrenzte und aus betriebswirtschaftlicher Sicht sinnvolle Funktionen. Sie werden entweder von einer Unternehmenseinheit oder durch externe Partner erbracht [7].
Die Prinzipien der Service-orientierten Architektur bezüglich der Bereitstellung, des Findens und des Kombinierens neuer Dienste leiten sich von dem sogenannten "Find-bind-execute"-Paradigma ab (siehe **Bild 1**). Dieses Paradigma beschreibt, wie ein Dienst (engl. Service) veröffentlicht, gesucht und genutzt werden kann. Zunächst muss ein Diensteanbieter den Dienst veröffentlichen. Dies geschieht, indem der Diensteanbieter eine Beschreibung des Dienstes in einem Verzeichnis hinterlegt. Diese Dienstebeschreibung definiert genauer, wie der Dienst aufzurufen ist und was der Dienst z.B. an Daten zurückliefert. Die Dienstebeschreibung ist somit eine vollständige Diensteschnittstelle. Ein Dienstekonsument kann nun eine Anfrage an das Verzeichnis stellen. Die Anfrage wird dadurch beantwortet, dass der Dienstesuchende die Dienstebeschreibung erhält. Somit kann der

Dienstesuchende letztendlich auch den Dienst aufrufen und nutzen. Weitere Grundprinzipien der SOA bezüglich Dienste sind:

- Wiederverwendbarkeit der Dienste
- Dienste müssen unabhängig von anderen Diensten sein (lose gekoppelte Dienste).
- Komponierbarkeit/Orchestrierbarkeit der Dienste
- Erreichbarkeit und Verfügbarkeit der Dienste
- Unabhängigkeit vom Ort der Dienstebereitstellung

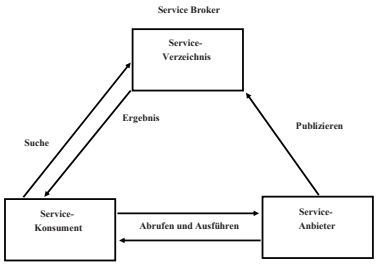


Bild 1 "Find-bind-execute"-Paradigma

Die SOA bietet also grundlegende Prinzipien zur Erfüllung der Anforderungen hinsichtlich des Bereitstellens, Findens und Kombinierens von Diensten. Wie dies in zukünftigen Telekommunikationsnetzen umgesetzt werden kann, wird in den folgenden Kapiteln näher erläutert.

3 SIP-basierter Peer-to-Peer-Ansatz

Im Rahmen des BMBF-Projekts "IMS- oder P2P-basierte Dienstebereitstellung und -entwicklung für kundenspezifische Kommunikationsprozesse (TeamCom)" wurde basierend auf einem Peer-to-Peer-Ansatz (P2P) das "Find-bind-execute"-Paradigma angewandt. Hierzu wurden verschiedene mögliche Peer-to-Peer-Lösungen bezüglich ihrer Anforderungen an Netze analysiert. Dabei ergab sich, dass ein teilzentrales Modell, basierend auf den SIP-Standards, gewählt werden sollte, da es gegenüber anderen Peer-to-Peer-Modellen erhebliche Vorteile bietet. Diese Vorteile sind z.B. das einfachere Erfüllen regulatorischer Anforderungen oder spezialisierte Suchmöglichkeiten. Dieses so genannte Hybrid-P2P-Modell zeichnet sich durch eine semizentrale Architektur, wie in **Bild 2** dargestellt, aus. Neue Dienste können in diesem Modell sehr leicht bereitgestellt werden [1] und darüber hinaus auch gesucht, gefunden und kombiniert werden.

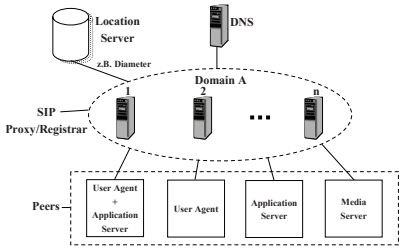


Bild 2 Hybrid P2P-Modell

Eine Reihe von SIP Proxy/Registrar Servern sind über die gleiche Domain mittels DNS (Domain Name System) [8; 9] erreichbar. Somit können sich SIP-Netzelemente (z.B. SIP User Agents) einfach an diesem P2P-Netz registrieren. Durch eine einfache Anbindung von Application-Servern und Media-Servern als Peers können Mehrwertdienste bereitgestellt werden.

4 Erweiterung des Peer-to-Peer-Ansatzes

Die Architektur aus **Bild 2** muss um einige Funktionen erweitert werden, um das Finden und Veröffentlichen von Diensten zu unterstützen. Im Laufe der letzten Jahre hat die IETF das Session Initiation Protocol und dessen Erweiterungen spezifiziert. Eine dieser Erweiterungen ist das SIP Event Framework [2; 3], welches eine Architektur zur Überwachung von Statuspublikationen (z.B. Presence) bietet. Unter Einbeziehung dieser Standards kann eine Peer-to-Peer-Dienste-Architektur erstellt werden

Die Veröffentlichung eines neuen Dienstes wird mittels der SIP-Nachricht PUBLISH realisiert. Der SIP-Body dieser Nachricht beinhaltet die Dienstebeschreibung [4]. Im Folgenden ist eine exemplarische SIP-Nachricht PUBLISH dargestellt.

```
PUBLISH sip:servicex.provider1@DomainA.de SIP/2.0
To: < sip:servicex. provider1@DomainA.de>
From: < sip:servicex. provider1@DomainA.de>;tag=1234
Expires: 6000
Event: x-serviceinfo
Content-Type: application/servinfo+xml
Content-Length: ...

<?xml version = "1.0"?>
<servinfo xmlns="urn:ietf:params:xml:ns:servinfo"
          version="0" state=full>
 <serviceID id="123"/>
 <keyword value="conference">
 <servicelocation>
   sip: servicex.provider1@DomainA.de
 </servicelocation>
```

```
  <mediatype>
    <audio direction="sendrecv">
    ...
  </mediatype>
</servinfo>
```

Die Dienstebeschreibung liegt in Form von XML vor und enthält neben einer für Menschen lesbaren Dienstebeschreibung auch weitere maschinenlesbare Informationen wie z.B. Schlüsselwörter (z.B. Konferenz oder Presence), Kommunikationsformen (z.B. Audio, Video, Text) und die Übertragungsrichtungen der genutzten Medienströme. Die PUBLISH-Methode wird an das Overlay-Netzwerk, bestehend aus SIP Proxy/Registrar-Servern und Discovery-Servern (auch Event Server genannt), gesendet [5]. Der SIP Discovery Server unterstützt das Veröffentlichen und Finden von Diensten. Alle Informationen, die zu den SIP Events (Diensteveröffentlichung, Dienstefindung) gehören, behandelt der SIP Discovery Server (siehe **Bild 3**). Der SIP Discovery Server speichert die Daten der Diensteveröffentlichungen z.B. in einem Pool von Datenbanken. Wird an den Discovery Server eine Anfrage (Dienstesuche) gestellt, durchsucht dieser die gespeicherten Daten und liefert die gefundenen Ergebnisse zurück. Diese Ergebnisse beinhalten neben einer Beschreibung des bzw. der Dienste(s) auch die URI (Uniform Resource Identifier) zum Abrufen des oder der Dienste(s).

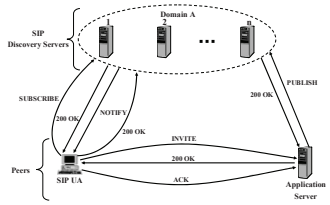


Bild 3 SIP/SOA-Architektur

Damit das Finden eines Dienstes ermöglicht wird, werden noch zwei weitere SIP-Nachrichten mit eingeführt, nämlich SUBSCRIBE und NOTIFY. Die Suchanfrage wird hierbei mittels der SIP-Nachricht SUBSCRIBE getätigt, welche hier exemplarisch dargestellt ist.

```
SUBSCRIBE sip:servicerequest@DomainA.de SIP/2.0
To: < sip: servicerequest @DomainA.de>
From: <sip:userA@DomainA.de>;tag=4567
Event: x-serviceinfo
Accept: application/servinfo+xml
Content-Type: application/servinfo+xml
Expires: 0
Content-Length: ...

<?xml version = "1.0"?>
<servinfo xmlns="urn:ietf:params:xml:ns:servinfo"
          version="0" state=full>
  <keyword value="conference">
</servinfo>
```

Die Antwort auf die Suchanfrage liefert die SIP-Nachricht NOTIFY, die hier auch beispielhaft dargestellt wird.

```
NOTIFY sip:userA@DomainA.de SIP/2.0
To: <sip:userA@DomainA.de>;tag=4567
From: < sip: servicerequest @DomainA.de>
Event: x-serviceinfo
Content-Type: application/servinfo+xml
Content-Length: ...
[Veröffentlichtes Dokument]
```

Im Folgenden wird die Dienstefindung näher beschrieben (siehe auch **Bild 3**).
Der Teilnehmer (hier: SIP UA) sendet eine SUBSCRIBE-Nachricht an das Overlay-Netzwerk. Diese SIP-Nachricht ist die Suchanfrage nach einem speziellen Dienst, der z.B. per Schlagwort gesucht wird. Der SIP Discovery Server empfängt nun die SUBSCRIBE-Nachricht und verarbeitet diese, indem er aus seinem Datenbestand den Dienst sucht, der zu dem Schlagwort passt. Daraufhin sendet der SIP Discovery Server eine NOTIFY-Nachricht an den Teilnehmer zurück. Diese Nachricht enthält die Informationen für den gesuchten Dienst. Da eventuell zu einem späteren Zeitpunkt neue Dienste veröffentlicht werden können (z.B. von anderen Peers), die auch zu dieser Suchanfrage passen, besitzt jede Suchanfrage eine Lebensdauer. Die Lebensdauer der Suchanfrage wird durch einen SIP-Header-Eintrag (Expires) kenntlich gemacht. Falls das Expires-Feld den Eintrag 0 trägt, wird die Suchanfrage beendet. Ein größerer Wert des Expires-Feldes legt die Gültigkeitsspanne in Sekunden fest [4; 6]. D.h., falls ein neuer Dienst innerhalb der Gültigkeitsspanne der Suchanfrage veröffentlicht wird und dieser z.B. zu dem Schlagwort passt, wird dem Dienstesuchenden unmittelbar eine neue Antwort durch den SIP Event Server zugesandt. Die neue Antwort (SIP NOTIFY) beinhaltet nun die Informationen zu dem neuen Dienst.
Die hieraus resultierende Architektur ist in **Bild 3** dargestellt. Diese Architektur lässt sich gut mit der des "Find-bind-execute"-Paradigmas vergleichen (siehe **Bild 1**). Auch hier ist ein sogenanntes Dienstedreieck zu erkennen, welches dem Application Layer zuzuordnen ist. Die im SIP-Overlay-Netzwerk befindlichen SIP Discovery Server repräsentieren hierbei das Diensteverzeichnis (Service-Verzeichnis). Der Diensteanbieter ist hier das bereitstellende SIP-Netzelement, nämlich der Application Server, und der Dienstekonsument ist hier der SIP User Agent (SIP UA).
Da die Dienstebeschreibung nur eine ständige SIP URI liefert, wird natürlich auch ein Call Control Layer zum Signalisieren benötigt, welches durch das SIP-Overlay-Netzwerk, bestehend aus den SIP Proxy/Registrar Servern (Call Server), realisiert wird. In

Bild 4 ist die prinzipielle Architektur eines NGN mit Dienstedreieck dargestellt.

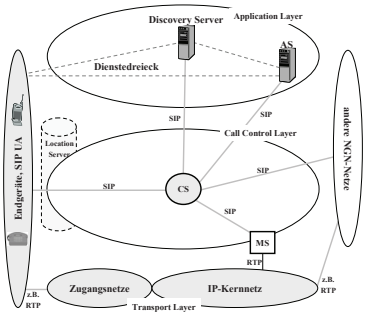


Bild 4 Architektur eines NGN mit Dienstedreieck

Im folgenden Abschnitt wird erläutert, wie eine einfache Dienstekombination erstellt werden kann.

5 Kombination von Mehrwertdiensten

Basierend auf dem bisher erläuterten Ansatz kann mittels einer einfachen Methode die Kombination verschiedener verteilter Dienste erreicht werden. Hierzu wird der Route Header [10; 11] beim Aufruf der Dienstekombination verwendet. Dieses SIP Header-Feld beinhaltet verschiedene SIP URIs, welche die Adressen der einzelnen Dienste darstellen. Anhand eines einfachen Beispiels soll die Anwendung der Dienstekombination veranschaulicht werden.
Ein beispielhafter Dienst könnte eine Audio/Video-Konferenz mit eingeblendetem Newsticker (z.B. aktueller Fußballergebnisse) sein. Die Audio-Konferenz, Video-Konferenz und der Newsticker wären hierbei einzelne (verteilte) Dienste. Jeder dieser Teildienste ist durch eine ständige SIP URI adressierbar (z.B. sip: audioconference.providerX@DomainA.de). Durch eine Verkettung der SIP URIs kann der gesamte Dienst realisiert werden. Der resultierende Dienst kann z.B. mit Hilfe der folgenden Teile einer SIP-Nachricht beschrieben werden.

```
INVITE sip: videoconference.providerZ@DomainA.de
SIP/2.0
...
Route: <sip: audioconference.providerX@DomainA.de;lr>,
       <sip: newsticker.providerY@DomainA.de;lr>
...
```

Die SIP-Nachricht (hier: INVITE) würde auf Grund des Route Headers gemäß **Bild 5** geroutet (Dienst 1 = Audio-Konferenz; Dienst2 = Newsticker; Dienst 3 = Video-Konferenz).

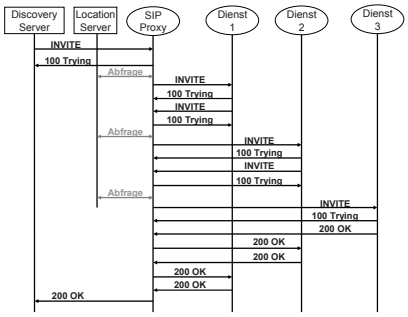


Bild 5 Aufruf einer Dienstekombination mit SIP

Wie in **Bild 5** dargestellt, würde die SIP-Nachricht INVITE von Dienst zu Dienst weitergereicht werden. Die Dienste selbst können hierbei Teile der SIP-Nachricht verändern. So müsste z.B. der Teildienst "Newsticker" den SIP Body abändern. Die Videodaten müssten nämlich nach der Bearbeitung durch die Video-Konferenz an den Newsticker weitergereicht werden, damit die Einblendung des Tickers möglich wird. Hierzu können einfach die entsprechenden Daten des SIP Bodys (hier: SDP (Session Description Protocol)), wie folgendes Beispiel zeigt, abgeändert werden.

```
...
v=0
o=client 1234 0 IN IP4 184.166.10.2
s=ServiceCombination
t= 0 0
c=IN IP4 184.166.10.2
m=audio 1000 RTP/AVP 0 8 3
m=video 2000 RTP/AVP 34
...
```

Die übermittelte IP-Adresse des c-Parameters beschreibt in Kombination mit dem Port des m-Parameters (hier: 2000 für Video) den Socket, auf dem die Videodaten vom Dienstenutzer erwartet werden. Damit der Newsticker die Videonutzdaten erhält, fügt dieser einfach einen neuen c-Parameter mit seiner IP-Adresse oberhalb des m-Parametes (Video) ein. Der Ausschnitt des SIP Bodys könnte dann wie folgt aussehen.

```
...
v=0
o=client 1234 0 IN IP4 184.166.10.2
s=ServiceCombination
t= 0 0
c=IN IP4 184.166.10.2
m=audio 1000 RTP/AVP 0 8 3
c=IN IP4 186.101.12.23
m=video 2000 RTP/AVP 34
...
```

Mittels dieser kleinen Änderungen innerhalb der SIP-Nachrichten können die entsprechenden Nutzdaten wie gewollt umgeleitet werden.
Dieses kleine Beispiel zeigt bereits, wie einfach neue Dienstekombinationen erstellt werden können. Die Vorteile dieses Ansatzes sind einmal die Nutzung aller Vorteile der Peer-to-Peer-Architekturen (Skalierbarkeit, Kostenreduktion, Fehlertoleranz) und die Nutzung von Standard-SIP-Netzelementen wie z.B. SIP User Agent und SIP Proxy/Registrar Server.

Eine Anwendung dieses neuen Ansatzes könnte die Umsetzung eines Dienste-Overlays für Mobilfunknetze sein. Hierbei könnten z.B. über Mobilfunk angebundene Endgeräte/Peers (z.B. Notebooks) eigene Dienste anbieten. Diese Dienste könnten darüber hinaus mit existierenden Diensten eines IMS kombiniert werden.

6 Literatur

[1] Lehmann, A.; Eichelmann, T.; Trick, U.: Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation. ITG-Fachbericht 208 Mobilfunk, Mai 2008

[2] Roach, A. B.: RFC 3265 – Session Initiation Protocol (SIP)-Specific Event Notification, IETF, 2002

[3] Niemi, A.: RFC 3903 – Session Initiation Protocol (SIP) Extension for Event State Publication, IETF, 2004

[4] Rahman et al.: Service Discovery using Session Initiation Protocol (SIP). United States Patent Application Publication, 2006

[5] Trossen, D.; Pavel, D.: Context Provisioning and SIP Events. International Conference on Mobile Systems, Applications, and Services (MobiSys2004), Boston, June 2004

[6] Trossen, D.; Pavel, D.: Service discovery & availability subscriptions using the SIP event framework. International Conference on Communications, IEEE 2005

[7] Tilkov, Stefan; Starke, Gernot: Einmaleins der serviceorientierten Architekturen. SOA-Expertenwissen – Methoden, Konzepte und Praxis serviceorientierter Architekturen. Heidelberg: dpunkt, 2007

[8] Mockapetris, P.: RFC 1034 – Domain Names – Concepts and Facilities. IETF, November 1987

[9] Mockapetris, P.: RFC 1035 – Domain Names – Implementation and Specification. IETF, November 1987

[10] Rosenberg, J.; Schulzrinne, H.; Camarillo, G.; Johnston, A.; Peterson, J.; Sparks, R.; Handley, M.; Schooler, E.: RFC 3261 – SIP: Session Initiation Protocol. IETF, June 2002

[11] Walker, S.: Implementation Agreement for SIP interface between Service Broker and Application Server. MultiService Forum, April 2005

Published in *Proceedings of the Fourth Collaborative Research Symposium on Security, E-learning, Internet and Networking (SEIN 2009)*, pp. 173-182, University of Applied Sciences, Darmstadt, Germany, ISBN: 978-1-84102-236-9

A new approach to classify and describe telecommunication services

A.Lehmann[1,2], W.Fuhrmann[3], U.Trick[1], B.Ghita[2]

[1] Research Group for Telecommunication Networks, University of Applied Sciences Frankfurt/M., Frankfurt/M., Germany
[2] Centre for Security, Communications and Network Research, University of Plymouth, Plymouth, United Kingdom
[3] University of Applied Sciences Darmstadt, Darmstadt, Germany
E-mail: lehmann@e-technik.org

Abstract

Service composition, especially the reusability of distributed Internet Protocol (IP) multimedia services, will play a decisive role in future telecommunication networks. Therefore present classifications and specifications for telecommunication services have been researched. The summarised results clearly illustrate the insufficiency of describing services from a consumer's view. So a new classification for telecommunication services is presented that will support customers to simplify the description of services. The aim of this approach is a simple specification and description language for reusable and distributed services which will support customers not only in describing services and service compositions. It should support customers to describe, find and use existing distributed telecommunication services.

Keywords

Multimedia, Value-added services, Service composition, Telecommunication

1. Introduction

Based on future networks, especially Next Generation Networks (NGN), and their service delivery platforms (SDP) the provision and composition of telecommunication services, also distributed telecommunication services, will be facilitated. In future customers will demand telecommunication services which will satisfy their requirements. To meet these increasing requirements, two of the critical steps are to implement the support for openness towards new services and to support the ability to describe services from a customer's view in future networks. This will lead to a number of challenges, such as classifying and characterising telecommunication services that will support customers to simplify the description of service compositions. Therefore present classifications and specifications for tele-communication services have been researched and are presented in the next chapter. The summarised results clearly illustrate the insufficiency of describing services from a consumer's view. A new classification for telecommunication services is presented in chapter 3 that will support customers to simplify the description of services. The aim of this approach is a simple specification and description language for reusable and distributed services which will support customers not only in describing services and service compositions. It should support customers to

describe, find and use existing distributed telecommunication services. Chapter 4 shows an example of a possible service description. Finally chapter 5 concludes the paper.

2. Existing definitions and classification models

Telecommunications (communications engineering) deals with capturing, processing, transmitting and storing information (communications). For this purpose telephony companies provide services (telecommunication services). The understanding of services differs in a broad range. The following lists different definitions and specifications for the notion of services in telecommunications.

- Services are functional properties of the network which support a certain form of communication between sources and sinks (P. Kühn, 1991).
- A service is a set of goods or valuable functions offered by a service provider to a customer (C. Abarca et al., 1997).
- An IP multimedia service is the user experience provided by one or more IP multimedia applications (ETSI TS 122.228, 2009).
- Services are made up of different service capability features (ETSI TS 122.105, 2008).
- A (Basic) telecommunication service is a term that is used as a common reference to both bearer services and teleservices (ETSI TS 122.105, 2008).
- IP multimedia services are the IP based session related services, including voice communications. IP multimedia sessions use IP bearer services provided by the PS (Packet Switched) CN (Core Network) Domain (ETSI TS 122.101, 2009).
- Multimedia services combine two or more media components (e.g. voice, audio, data, video, pictures) within one call. A multimedia service may involve several parties and connections (different parties may provide different media components) and therefore flexibility is required in order to add and delete both resources and parties (ETSI TS 122.101, 2009).
- Multimedia services are typically classified as interactive or distributed services (ETSI TS 122.101, 2009) (based on service classes specified by ITU-T (ITU-T Recommendation I.211, 1993)).

The listing above demonstrates that there are different meanings to services given by the standardisation bodies. They are basically divided into different bearer services. A bearer service is a type of service that provides the capability of transmission of signals between access points (ETSI TS 122.105, 2008). Bearer services are typically categorised by their information transfer characteristics, methods of accessing the service, interworking requirements (to other networks), and other general attributes. Information characteristics include data transfer rate, direction(s) of data flow, type of data transfer (circuit or packet) and other physical characteristics (Harte, J. et al, 1997). These bearer services are the basis for teleservices and supplementary services. A teleservice is a type of telecommunication service that provides the complete capability, including terminal equipment functions, for communication between users according to standardised protocols and transmission capabilities established by agreement between operators (ETSI TS 122.105, 2008). A supplementary service is a service which modifies or supplements a basic

telecommunication service. Consequently, it cannot be offered to a user as a standalone service. It shall be offered together with or in association with a basic telecommunication service (ETSI TS 122.105, 2008).

Based on these definitions a telecommunication service is a combination of a bearer service and a teleservice as shown in Table 1 (ETSI TS 122.001, 2009), (ITU-T Recommendation I.210, 1993).

telecommunication services	
teleservice	
basic teleservice	basic teleservice + supplementary service
bearer service	
basic bearer service	basic bearer service + supplementary service

Table 1: Categorisation of telecommunication services

There are two further definitions for services in telecommunications based on bearer services. These are the IP multimedia services and so-called value added (non-call related) services. The value added services include a large variety of different operator specific services/applications. They are usually not specified by 3rd Generation Partnership Project (3GPP). The services can be based on fully proprietary protocols or standardised protocols outside 3GPP (ETSI TS 122.101, 2009). Figure 1 gives an overview of all the described definitions.

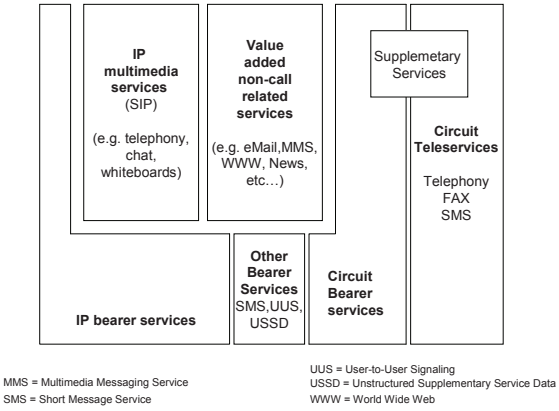


Figure 1: Service classification

The classification of services and applications is done according to ITU-T Recommendation I.211 (ITU-T Recommendation I.211, 1993), (Velez, J. et al,

2000). An (communication) application is defined as a task that requires communication of one or more information streams between two or more parties that are geographically separated. More specifically, an application can be characterised by the following main attributes (Kwok, T., 1995), (Velez, J. et al, 2000):

- intrinsic time dependency (time or non-time-based)
- delivery requirements (real-time or non-real-time)
- directionality (unidirectional or bi-directional)
- symmetry of the communication (symmetric or asymmetric)
- interactivity
- number of parties

A set of applications with similar characteristics, or even a single application, can be classified as a service. According to ITU-T I.211, services can be classified as interactive or distributed (ITU-T Recommendation I.211, 1993).

Interactive services are typically subdivided into conversational, messaging and retrieval services (ETSI TS 122.101, 2009):

- Conversational services are real time (no store and forward), usually bi-directional where low end to end delays and high degree of synchronisation between media components are required. Video telephony and video conferencing are typical conversational services.
- Messaging services offer user to user communication via store and forward units (mailbox or message handling devices). Messaging services might typically provide combined voice and text, audio and high-resolution images.
- Retrieval services enable a user to retrieve information stored in one or many information centres. The start at which an information sequence is sent by an information centre to the user is under control of the user. Each information centre accessed may provide a different media component, e.g. high resolution images, audio and general archival information.

Distributed services are typically subdivided into those providing user presentation control and those without user presentation control (ETSI TS 122.101, 2009):

- Those without user control are broadcast services where information is supplied by a central source and where the user can access the flow of information without any ability to control the start or order of presentation e.g. television or audio broadcast services.
- Those with user control are broadcast services where information is broadcast as a repetitive sequence and the ability to access sequence numbering allocated to frames of information enables the user to control the start and order of presentation of information.

Up to now services were classified in respect of providers or networks. This hinders differentiation of services, because they were not classified regarding to service components/media types (e.g. audio, video, data) but rather in dependency to the needed resources. The problem of classifying services by the presented methods is that there is no chance to link up services. The next chapter presents a new method to classify services in telecommunications, which will allow services to be concatenated.

3. New approach to classify multimedia services

Services in telecommunications always offer different media and data. Media is generally known as "A general term of means and techniques for spreading information (incl. entertainment, music et al.)" (Meyers Großes Taschenlexikon, 1987). Media consists of different media types. These media types are divided into media perception and media presentation.

Media perception describes the modality of how humans will conceive information. The information humans may conceive are joint to their sensory perception. Therefore only the physiological perception is considered and no subjective or other natured scoring. Examples of acoustic media are music, speech and sound. Visual media are text, still image or video. The presentation of media belongs to the processing of information e.g. the encoding of video (Boles, D. et al., 1996).

A media type will be described through its presentation and perception of media. It will be named by its media perception (e.g. audio, video, text). Media types can be time-invariant or continuously. Time-invariant media types are for example text and still image and continuous media types are for example audio, video, animation (Boles, D. et al., 1996). The data of multimedia can be processed by the following components:

- Sources output data streams of concrete media types.
- Sinks receive data streams and they can potentially display them.
- Filters combine the characteristics of sources and sinks. Filters are mostly used to convert data streams (e.g. mono-to-stereo).

These three components are called media objects. All media objects are connected by media channels/ports which belong to one media type. These ports represent the I/O-interfaces of a media object. Figure 2 shows an overview of the media objects.

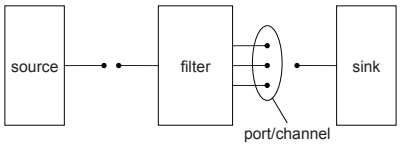


Figure 2: Media objects

These elements are well known from telecommunications. In the near future telecommunication services will support more media types, which can be derived from the human senses. All five senses are shown in the following listing.

- Auditiv (Hearing)
- Visual (Sight)
- Gustatory (Taste)
- Olfactory (Smell)
- Haptic (Touch)

Four further senses (pain, balance, proprioception and temperature) are known, which will be assigned to the sense touch, only for simplification. Furthermore there are also media types which are not sensually imperceptible. These media types can only be processed by machines. Following they will simply be called data. Consequently there are six different media types. They could be characterised more precisely (e.g. Hearing: Audio, Speech or Sound). Table 2 gives an overview of the media types and some of their possible characteristics.

Media type	**Characteristics**
Hearing	sound, speech, music
Sight	video, animation, text, still image, light
Taste	sweet, sour, bitter, spicy, umami
Smell	camphorous, fishy, malty, minty, musky, spermous, sweaty, urinous
Touch	pressure, temperature, balance, proprioceptive
Data	file, sensor, actuator, trigger, information

Table 2: Media types and their characteristics

The shown classification of media objects and types creates a possibility to describe compositions of multimedia streams. Media compositions are divided into the three type's spatial, temporal and configurable compositions (Steinmetz, R., 1993). Spatial composition describes the geometrical structure of the presentation of the media and will not describe the way a service should works. The configurable composition describes dynamic connections between the single media objects (e.g. how text has to be converted into speech) and the temporal composition describes temporal relations of the media objects. The relevant media composition for telecommunication services is the temporal composition combined with a configurable composition called service function (see Figure 7). According to Allen (Allen, James F., 1983) relations between two intervals can be divided into 13 different relations. Figure 3 shows seven of them. All other relations result in inverted relations. The only relation without inversion is "equals". All these relations could also be described by a hierarchy (see Figure 4 for an example).

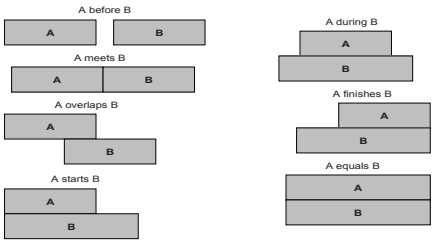


Figure 3: The thirteen possible relationships

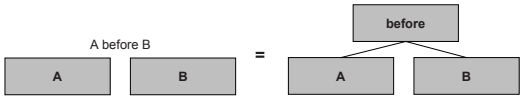


Figure 4: Transformation to hierarchical representation

With the presented methods it is still possible to describe multimedia services. But this description is very vague. To specify the description the media objects which are bound to media types, which maybe specified by their characteristics, are extended with inner methods. The inner methods are derived from the possible basic connection types in telecommunications (see Figure 5).

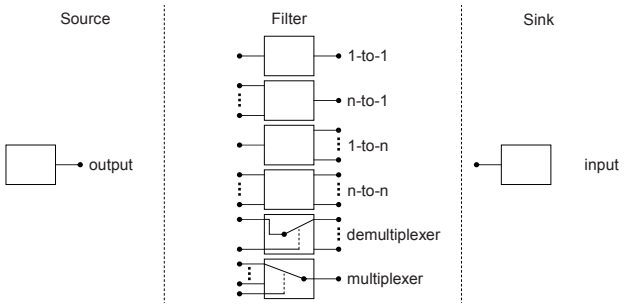


Figure 5: Basic types of media objects

The shown basic types of the filter elements are sorted into three groups, intra-media, inter-media and special types. The intra-media filters will only process one media type. Handling of different media types will be processed by the inter-media filters. The special types can be used inter or intra media. Figure 6 presents an overview of the different filter types.

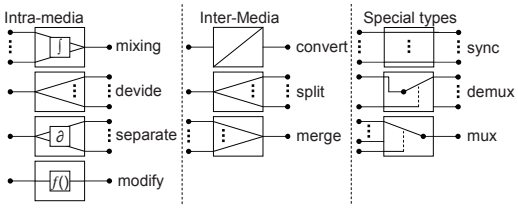


Figure 6: Filter types

The next chapter will give you an impression of how to describe telecommunication services by using the defined elements and methods presented in this chapter.

4. Exemplary service description

The following example describes a simple service scenario. Some information (e.g. the identity of a subscriber) will be displayed to another subscriber, if the information matches to a simple result of a function (e.g. the initiating subscriber's name is "bob"). Afterwards the two subscribers may have a multimedia session to communicate with each other (e.g. audio and video telephony). How could this simple service be abstracted by the usage of the elements and methods presented in the previous chapter?

First it is essential to distinguish two different types of descriptions. The first one is the service function description and the second is the temporal service composition description. Both are shown in Figure 7.

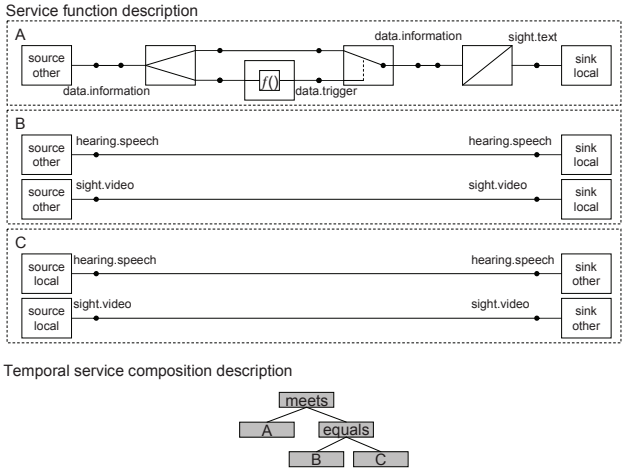


Figure 7: Exemplary service description

The service function description in Figure 7 depicts three different service functionalities. The first one "A" is the functionality which describes that some information is transmitted and converted to text, if the information triggers the media stream containing the information. The descriptions "B" and "C" describe two media streams, namely speech and video. All three functional descriptions are temporarily composed by the hierarchical tree in Figure 7. This can be transformed to the following view according to Allen (Allen, James F., 1983) (see Figure 8). Figure 8

clearly depicts clearly that the functionality "A" is followed by "B" and "C" and that "B" and "C" will exist for exactly the same time.

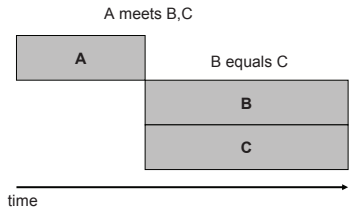


Figure 8: Temporal description with timeline

5. Conclusion

The presented classification for telecommunication services illustrates that service description and composition could be handled easily. The new classification provides tools to support customers to describe telecommunication services themselves. The created service description could also be transformed into a technical representation such as the extensible markup language (XML). The shown model is as well extendable, e.g. the inner methods of the filter types could be more specified (e.g. modify.loudness).

The next steps to be taken are the verification of the presented model and a conversion into a technical representation (e.g. XML). Also the attempt to convert the technical representation to a mathematical correspondent is planned to make the entire model may be provable.

6. Annotation

The research project providing the basis for this publication was partially funded by the Federal Ministry of Education and Research (BMBF) of the Federal Republic of Germany under grant number 1704B07. The authors of this publication are in charge of its content.

7. References

Abarca, C., Farley, P., Forslöw, J. Garcia, J.C., Hamada, T. Hansen, P.F., Hogg, S., Kamata, H., Kristiansen, L., Licciardi, C.A., Mulder, H., Utsunomiya, E., Yates, M. (1997), "Service Architecture Version 5.0", TINA-C

Allen, James F. (1983), "Maintaining knowledge about temporal intervals", Communications of the ACM, ACM

Boles, D., Becker, A., Bley, S., Dauelsberg, M., Eßer, A., Knoblich, C., Kölling, M., Logemann, D., Mertins, G., Prusch, T., Steen, B., Unbehaun, S., Voßkamp, R. (1996),

"Zwischenbericht der Projektgruppe Multimedia-Präsentation im Gesundheitswesen", Universität Oldenburg - Oldenburg University

ETSI TS 122.001 V8.0.0 (2009), Technical Specification, "Principles of circuit telecommunication services supported by a Public Land Mobile Network (PLMN)", ETSI

ETSI TS 122.101 V9.3.0 (2009), Technical Specification, "Service aspects; Service principles", ETSI

ETSI TS 122.105 V8.4.0 (2008), Technical Specification, "Services and service capabilities", ETSI

ETSI TS 122.228 V8.6.0 (2009), Technical Specification, "Service requirements for the Internet Protocol (IP) multimedia core network subsystem (IMS); Stage 1", ETSI

Harte, J. Lawrence, Hoenig, M., McLaughlin, D., Kikta, R. (1997), "CDMA IS-95 for Cellular and PCs: Technology, Economics, and Services", McGraw-Hill - Telecommunications

ITU-T Recommendation I.210 (1993), "Principles of Telecommunication Services supported by an ISDN and the means to describe them", ITU-T

ITU-T Recommendation I.211(1993), "B-ISDN Service aspects", ITU-T

Kühn, P. (1991), "Vorlesungsskript Nachrichtenvermittlung I und II", Universität Stuttgart - Stuttgart University, Institut für Nachrichtenvermittlung und Datenverarbeitung

Kwok, T. (1995), "A vision for residential broadband services: ATM-to-the home", Proceedings of the Second International Workshop on Community Networking, IEEE

Meyers Großes Taschenlexikon, Band 14 (1987), Meyers Verlag

Steinmetz, R. (1993), "Multimedia - Technologie: Einführung und Grundlagen", Springer-Verlag

Velez, J, Correia, M. (2000), "Classification and Characterisation of Mobile Broadband Services", 52nd Vehicular Technology Conference, IEEE

Published in *Fourth International Conference on Internet and Web Applications and Services (ICIW)*, Venice, Italy, IEEE, ISBN: 978-1-4244-3851-8

TeamCom: A Service Creation Platform for Next Generation Networks

A. Lehmann, T. Eichelmann, U. Trick
Research Group for Telecommunication Networks
University of Applied Sciences
Frankfurt/Main, Germany
e-mail: {lehmann, eichelmann, trick}@e-technik.org

R. Lasch, B. Ricks, R. Tönjes
Research Group for Mobile Communications
University of Applied Sciences
Osnabrück, Germany
e-mail: {r.lasch, b.ricks, r.toenjes}@fh-osnabrueck.de

Abstract— The development of value added services is currently still very time and cost consuming. The need for specific user generated and in particular business-to-business services demands for efficient service development methods. This paper presents a service creation environment that supports the application developer to compose a service based on reusable components and to describe the business process through a control logic. Forthe service description a language that has been optimized for business prozesses is suggested: the Business Process Execution Language (BPEL). However, BPEL has not been developed for control of specific, in particular real time, communication services in heterogeneous networks.Therefore the paper presents a parser translating the business process description into Java code and supporting the deployment of the service in a service execution environment based on JAIN SLEE. The provided elementary communication Service Components hide the underlying heterogeneous communication networks. Thereby the developer does not need any detailed knowledge of communication protocols and is able to focus on the application logic instead. This leads to new opportunities for rapid and efficient service creation using a new Service Creation Environment (SCE) with higher level of abstraction and automated service generation.

Keywords-component; SCE (Service Creation Environment; NGN (Next Generation Networks); Service Components; JAIN SLEE

I. INTRODUCTION

The provision of customer specific communication processes is currently still very time and cost consuming. As a consequence the penetration of specific multimedia services is low. This holds for business-to-business (B2B) services as well as for individual services, which are even in the Web 2.0 area limited to simple services. Hence many capabilities of today's multimedia networks remain often unexploited.

B2B services offer a high potential to allow for the acceleration of processes and workflows within and between organizations. Unfortunately, the development of mobile B2B services requires still a lot of detailed knowledge about mobile communication systems and their protocols. Moreover the application developers need a deep understanding of the embedded business processes. The wide range of the necessary knowledge hinders the growth of mobile B2B services and fosters proprietary solutions. Therefore the TeamCom project [1] aims at enabling fast, easy and cost efficient provisioning of value added services, in particular B2B services, by creating a new high level Service Creation Environment (SCE). Elementary communication Service Components are derived from the requirements for telecommunication services. To support the service developer these communication components should be provided and integrated into a generic Service Creation Environment for mobile B2B services in heterogeneous networks.

For the orchestration, i.e. composition, of services various SCE have been suggested recently in the literature. As lessons learnt the European SPICE [2] and OPUCE [3] projects point out the need for a clear separation between professional and end-user service creation. The LOMS [4], the MAMS [5] project and the Open Source Initiative SPAGIC [6] have proven the advantages of a graphical service creation workbench, developed in particular for small and medium sized enterprises, to ease B2B service creation. To cope with evolving service requirements and the heterogeneity of the underlying communication and computing infrastructure, respectively, the SeCSE [7] and the PLASTIC [8] project suggested self-adapting service oriented applications.

This paper takes a systematic approach for service creation by reusing communication Service Components and suggests building the service creation environment on top of the future control layer of next generation networks, the IP Multimedia Subsystem (IMS) [9], which is based on the Session Initiation Protocol (SIP) [10]. For the first time, IMS offers the opportunity to provide services uniformly over heterogeneous networks (cellular networks, WLAN and fixed networks) and outside of the user's home network. It provides an integrative support for mobility, quality of service, security and service dependent accounting. In contrast to this centralized approach peer to peer (P2P)

communication, e.g. Skype and P2P-SIP [11], could provide a cost efficient and scalable alternative.

To speed up the development of services, in particular of B2B services, this paper proposes a fourfold approach:

- Abstraction of heterogeneous network interfaces
- Definition of reusable communication Service Components
- Development of a Service Creation Environment combined with automated service generation employing reusable components
- Use of partial services by cooperating application servers

The reminder of this paper is structured as follows: The next chapter presents the overall architecture of the system. Chapter III describes the generic Service Creation Environment and chapter IV discusses the elements for supporting the service life cycle as a whole. Chapter V shows an example of service creation and deployment evaluating the approach and chapter VI finally concludes the paper.

II. ARCHITECTURE

The TeamCom Project proposes an integrated architecture for service creation, deployment and execution (see Fig. 1). This architecture can be divided into four layers: (1) The Service Creation Environment (SCE), which includes the design and composition of new services, (2) the Service Deployment (SD), (3) the Service Execution Engine (SEE) containing one or more Application Servers (AS) based on JAIN SLEE (JAIN Service Logic Execution Environment) and (4) the Service Transport Layer (STL). The Service Transport Layer abstracts different protocols in order to enable upper service layers to be independent of a specific communication protocol. Therefore it supports several communication networks, e.g. IP Multimedia Subsystem or Peer-to-Peer SIP. The Service Creation Environment contains a GUI, permitting a developer to orchestrate Service Components and to integrate external services easily without having to develop low-level software code. The Service Execution Environment establishes a layer where created Services Components are deployed and activated.

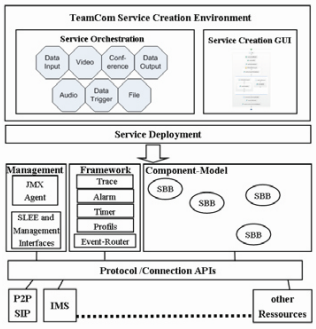


Figure 1. TeamCom Architecture

In order to support Next Generation Networks that are based on IP, different software architectures for the Service Execution Environment have been evaluated. The following requirements for a Service Execution Environment have been defined: independence from the operating system, service orchestration based on components, comfortable component deployment, support for SIP, extensibility for other protocols and possibility for co-operation between numerous application servers. To fulfil these requirements the architecture makes use of the JAIN SLEE standard [12].

The JAIN SLEE standard defines a component, event and transaction based architecture. The architecture is written in the Java language and standardised by a Java Community Process. It is part of the JAIN (Java API for Integrated Networks) Initiative consisting of several telecommunication companies. JAIN SLEE is designed for ensuring low latency and providing high throughput to accomplish the requirements for communication services. It uses a distributed component model similar to Enterprise Java Beans (EJB) [13]. Therefore it is often referred as "EJB for Communications".

In the standard so called Resource Adaptors (RA) are defined abstracting the underlying infrastructure. These Resource Adaptors provide a common Java API which hides the communication protocol underneath. In detail when a communication protocol message is received the corresponding RA translates this message into a Java event class. Afterwards the event and an activity class both together are passed to the JAIN SLEE event router. A JAIN SLEE activity represents a session of a communication protocol e.g. a SIP dialog. The event router utilizes these objects to look up the services which requested to receive the specific event. Accordingly the service itself is able to react on the event and to create an answer by using defined Java Interfaces. The answer is translated to a communication protocol response by the RA. The JAIN SLEE standard pre-defines (among others) the interface for a SIP Resource Adaptor. Additional Resource Adaptors can be added manually. In TeamCom several Resource Adaptors for email

communication (Mail RA), for evolved Web Service access (Web Service RA), for security mechanisms based on TLS (Transport Layer Security) (TLS RA) and IMS extensions for SIP (IMS RA) have been developed. Hence a service based on JAIN SLEE is able to support heterogeneous networks.

The service itself is composed of one or more Service Building Blocks (SBB). These SBBs contain the application/service execution logic and are deployed on a JAIN SLEE Application Server such as Mobicents [14]. The SBB component model includes a lifecycle, registration and security management. In addition, SBBs are able to access timer, trace, alarm and profile facilities which are also provided by a JAIN SLEE server.

III. SERVICE CREATION ENVIRONMENT

As depicted in the last chapter the Service Creation Environment includes service orchestration on the basis of reusable Service Components and existing services. In addition to the components a service logic, similar to "if then" statements, is required for describing the workflow properly. The following two sections explain both.

A. Reusable Service Components

Several elementary Service Components are derived from the requirements for telecommunication services. By combining these elementary components it should be possible to describe and generate other valuable services. These elementary Service Components comprise audio, video, text, file, conference, data input, data output and data trigger components. This chapter discusses the abstract Service Components in detail.

Audio: The Audio Component handles all kind of audio communication including the establishment of a call, answering calls, manipulation of audio streams (e.g. mixing, transcoding) and sending and receiving DTMF tones.

Video: The Video Component is responsible for playing and recording of video streams. It enables to create and close video calls and to combine different video signals for merging a new video stream.

Text: This component exchanges messages between two partners and has the capabilities of handling strings, e.g. search for a specific word in a text, replace alphabetic characters or change the encoding of a text.

File: The File Component handles creation, deletion, sending and receiving of binary files. Another task of File is to write and read any kind of data from and to any position in a file. Finally this component is able to rename files or directories.

Data Input: All kind of data queries are processed by the Data Input component. This includes database queries as well as reading data from a sensor.

Data Output: The counterpart of Data Input is Data Output being concerned with writing data to a destination e.g. to a database or to control an actuator.

Conference: This is a special kind of communication component because it re-uses internal functions of the previously described components, e.g. audio stream mixing. On top of this, the Conference Component provides functionalities for creating and deleting conference "rooms", adding and removing users to/from a room.

Data Trigger: The Data Trigger is closely related to an event generator. If a specific data trespasses a value an event is triggered. This data can be a sensor value, a timestamp or periodical dates.

B. Service Logic

Business workflows within and in between enterprises usually follow defined processes, the so called business processes. This resulted in the definition of the BPEL (Business Process Execution Language) standard [15], an XML [16] based process description language, building completely on web services.

Most often BPEL is used to orchestrate web services but the language is protocol independent. It is possible to create so called bindings which could specify the usage of BPEL for other protocols than SOAP (Simple Object Access Protocol) [17]. In BPEL it is possible to define synchronous and asynchronous processes. Thus it fits very well to be used in combination with JAIN SLEE. For a structured process different BPEL tags are defined, e.g. sequence, if, while. Moreover forked processes can be created, allowing for execution in parallel and synchronisation afterwards. As depicted in Fig. 2 the BPEL process has the ability to 'invoke' other services asynchronously after having received a request from a client. The 'reply' BPEL element is used to send an answer to the client.

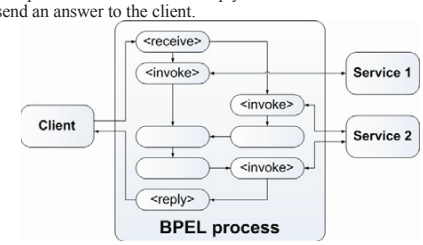


Figure 2. BPEL Process Example

C. Service Creation

To orchestrate value added services easily a graphically interface is provided by the TeamCom Service Creation Environment. It is possible to drag and drop Service Components to the workspace where the BPEL process is designed. Additionally BPEL elements can be dragged into the workspace and be connected to the Service Components. TeamCom employs BPEL to compose and control the sequence of the Service Components. All defined Service Components are provided as partner links in BPEL. Partner links define the communication partners and their roles in a process. Our Service Creation Environment only relies on a subset of all BPEL commands and doesn't require a BPEL engine. So it could be possible to choose another language for the TeamCom SCE to describe a service then BPEL because it is only used to define the service logic. Literature has shown that other service description languages are often

to protocol specific and/or not extendable (e.g. Call Processing Language) [18] [19] or they were withdrawn (e.g. Service Control Markup Language) [20].

After configuring all elements and components it is possible to create a service which can be deployed on an Application Server.

The Service Components are described in the Web Services Description Language (WSDL) [21] which is also based on XML. Therefore all components provide an interface independent of a communication protocol. In WSDL the methods of the Service Components are defined e.g. the 'onCall' method of the Audio Component. It is possible to support different implementations of Service Components, depending on the desired communication system. If a service should be deployed in an IMS network it is necessary to implement an Audio Component which is capable of the IMS SIP extensions. Thus every implementation has to create a WSDL binding, which describes the mapping between the abstract WSDL operation and the implementation.

The interfaces of a BPEL process itself are also described by WSDL. Therefore it is possible to include an external BPEL process in a newly created BPEL process. As a result small, simple and more generic services could be created which are reusable in other services. Both the integration of external services and Service Components is carried out in the same way.

IV. Service Life Cycle

The ability to manage the lifecycle of B2B services is fundamental for achieving success within mobile service platforms. The service life cycle can be subdivided into a succession of operations that include service creation, deployment and execution. The following section introduces the selected Service Execution Environment before deriving the associated creation and deployment process.

A. Service Execution Engine

The B2B services require a Service Execution Engine that can conduct the business processes on top of the telecommunications infrastructure. The question rises how these two worlds can be combined.

In telecommunications a plethora of protocols and standards have to be supported (here in particular SIP and IMS). In such a heterogeneous environment, the services should be decoupled from the various underlying communication networks. The JAIN SLEE standard provides resource adaptors that abstract from the various underlying networks.

Complementary BPEL engines allow executing BPEL process instances. For this the BPEL processes are deployed on the BPEL engine using engine specific deployment information.

Four different approaches were analysed for accessing the BPEL information within the JAIN SLEE, which are depicted in Fig. 3:

1. Master SBB reads an XML file (e.g. BPEL).
2. Access of BPEL engine over resource adaptor.
3. Access of BPEL engine over J2EE connector.
4. Translation of BPEL into JAIN SLEE logic

The direct use of XML files is quite static. Also the files must be parsed during the execution time which means lower performance. In contrast alternative two and three would keep the BPEL engine allowing to run the workflows as BPEL processes. In alternative two BPEL would be treated like a protocol and invoke an event over the resource adaptor [22]-[24]. Alternative three employs the J2EE Connector Architecture (JCA) [25] providing via EJB (Enterprise Java Beans) an interface to the Java based BPEL engine operating in a J2EE environment. However, both alternatives with external BPEL engines limit the real time capabilities of the service. Also the usage of a BPEL engine will not fit to a P2P model, because the peers can not benefit by the BPEL engine.

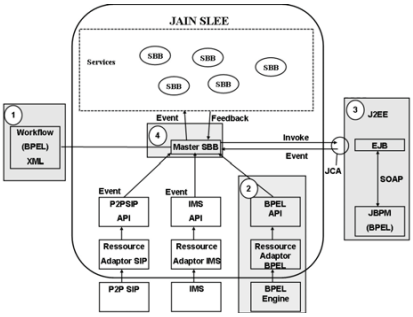


Figure 3. Integration of BPEL and JAIN SLEE

Therefore the fourth alternative was chosen, that translates the resulting BPEL files into a JAIN SLEE service. This means that the service which will be created is designed with a graphical editor based on BPEL elements. Within this solution no BPEL engine is used during the execution of the service, which will fasten up the execution time and also support P2P networks. Please note that in this case the Service Components can be integrated with the control information in only one (or a few) SBB, providing the service. This approach is detailed in the next section.

The SEE may provide different services, i. e. specific business processes, in parallel. Moreover, the B2B service may use external service parts that are distributed over different Application Servers.

B. Service Creation and Deployment

Service creation and deployment can be realised in five steps: writing a non-technical description of the service, converting this description to a service description language, analysing the description language, generating Service Building Blocks from the description language and deploying the service. First a business description has to be verbalised. The description could be e.g. a text, depending on the process and the involved people. Afterwards a service developer is able to create a BPEL based description

graphically via the service creation Graphical User Interface (GUI). In this step the service developer utilizes the TeamCom Service Components which are included in BPEL as partner links and configures their input variables e.g. video sender address. After finishing the BPEL process description the generation of the service will start (see Fig. 4).

The service designed by the service developer shall not be executed on a BPEL process engine. Instead the service shall run on a JAIN SLEE application server. So the BPEL process has to be converted in a form to be executable as a JAIN SLEE service. The code generator analyses the BPEL process and parses the workflow step by step. The result from each individual step is saved in a XML service document that could be described as an intermediate language for a JAIN SLEE service. Finally the goal of the code generator is the creation of the Java classes and the necessary descriptor files needed for a JAIN SLEE service.

While the code generator parses the BPEL process, it analyses the BPEL activities. Pending on the BPEL activity the code generator adds pre-defined XML fragments to the XML service document. Process activities which initiate events for the partners or waiting for events from them must be examined to figure out which Service Component is affected by this event. For each method which can be invoked on a partner a pre-defined XML fragment in a XML fragment pool exists. If the Service Component is identified, an appropriate XML fragment which represents the used method from the BPEL process can be chosen from the XML pool and inserted into the XML service document. Other workflow activities perhaps need variables or data structures which are defined in BPEL. These structures are represented in XML schemata and have to be transformed into Java code also.

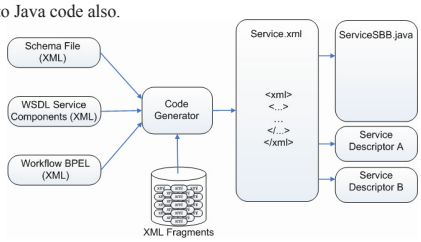


Figure 4. Code Generation

In Table I some examples for BPEL activities and their correspondent part in Java are shown. Now the XML service document contains all information necessary to generate the Java code, the service descriptor files and the build file.

The code generator will not create an intermediate language in future versions which will result in better performance. Instead of the XML fragments Java templates will be used and the necessary Java classes, a build file for the deployment and the descriptor files will be generated immediately.

TABLE I. COUNTERPARTS OF BPEL ACTIVITIES AND JAVA

BPEL Activity	Java
if	if()
while	while()
forEach	for(;;)
repeatUntil	do{}while()
wait	SLEE timer
sequence	sequential flow
flow	parallel flow
invoke	bidirectional method
receive	listener, getter method
reply	setter method
assign	operations e.g. String operations

.

C. *Evaluation of Service Creation*

The described service creation process has to be evaluated to validate the code generator. Therefore different scenarios were estimated. Those scenarios cover a maximised set of telecommunication services. The following enumeration is a subset of services to be evaluated:

- Session based services (e.g. active, passive, parallel or sequential sessions)
- Conferences (e.g. audio and/or video)
- Call-control (e.g. forwarding, blocking, forking)

These scenarios will be designed with the BPEL editor and generated as described before. The resulting Java source code has to be verified. So the evaluation will be an empirical process.

V. EXAMPLE

To demonstrate the TeamCom Architecture an exemplary and simple wake-up service has been developed. The Text Component is used to receive messages from a SIP client. These messages contain the date when the user will be woken and a text phrase the user wanted to receive. The application processes the incoming messages and passes the date to the Data Trigger Component. The timer will trigger the service at the reached date and finally our service creates a wake-up SIP MESSAGE which will be sent back to the user.

Fig. 5 depicts the creation of this wakeup service in the TeamCom SCE workbench showing on both sides the Service Components embedded in the service. The Text Component is responsible for receiving and sending messages. The Data Trigger Service Component is used for generating a timer.

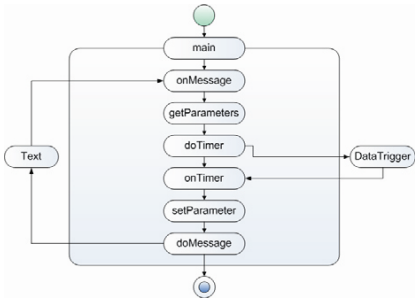


Figure 5. Wake-up Service Example

The main sequence contains the entire description of the service. First the service is waiting for an incoming text message from a client, which is handled by the 'onMessage' element of the Text Component. The received text message contains the date and text for the wake-up call. In 'getParameters' the BPEL assign element is used to get the necessary data for the timer. Afterwards the invoke element 'doTimer' is used to generate a timer via the Data Trigger Component. The element 'onTimer' will start the service again after the timer expired. The next element 'setParameter' is used to set the text phrase for the following element 'doMessage'. The reply element 'doMessage' is used to generate the wake-up SIP MESSAGE which will be sent back to the originating user.

VI. Conclusion

The development of new services for telecommunication applications and other IT systems is the most important means of innovation for telecommunication service providers. In particular, B2B applications possess a high potential to accelerate processes and workflows within and between organisations. Competition demands for faster innovation cycles to speed up time to market.

The presented SCE empowers small and medium sized enterprises to develop their B2B services in a time and cost efficient way. A graphical user interface supports even inexperienced developers to orchestrate the reusable components for value added B2B services, exploiting the full power of nowadays multimedia networks through predefined resource adaptors. The possibility for automated creation of communication services without a detailed knowledge about the used protocols and networks is given. The presented example was already created and generated by the usage of our SCE. Also different variations of this exemplary service could be developed within minutes. Service creation could be as easy as current website development and applicable for more people. Vice versa the ease of service creation will foster a better degree of utilisation of the telecom munication's multimedia networks. In the TeamCom architecture elementary communication Service Components are abstracted for a general use in possible Next Generation Networks. The Service Component interface is extensible for evolving protocols and networks.

Acknowledgment

This work has been supported by the German Federal Ministry of Education and Research (BMBF) under grant 1704A07 and 1704B07 in the TeamCom project.

References

[1] TeamCom project website: http://www.ecs.fh-osnabrueck.de/teamcom.html
[2] O. Droegehorn et al.: "Professional and End-User driven Service Creation in the SPICE platform", IEEE WOWMOM 2008, New Port Beach, California, USA, 23-25 June 2008.
[3] J. C. Yelmo et al.: "A User-Centric service creation approach for Next Generation Networks", ITU-T Kaleidoscope Event: Innovations in NGN - Future Network and Services, Geneva, 12-13 May 2008.
[4] J. Keiser, T. Kriengchaiyapruk. "Bringing Creation of Context-Aware Mobile Services to the Masses". In: IEEE SOA Industry Summit (SOAIS 2008), Hawaii, USA, July 2008.
[5] B. Freese, U. Staiger, H. Stein: "Multi-Access Modular-Services Framework - Whitepaper", Deutsche Telekom Laboratories, Berlin, June 2007.
[6] SPAGIC web site: http://spagic.org/ecm/faces/public/guest/home/solutions/spagic
[7] L. Baresi, E. Di Nitto and C. Ghezzi, "Toward Open-World Software: issues and challenges", IEEE Computer, Volume 39, No. 10: 36-43, October 2006.
[8] M. Autili et al.: "A Development Process for Self-adapting Service Oriented Applications", in proceeding of ICSOC 2007(442-448), Vienna, Austria, September 2007.
[9] TS 23.228: IP Multimedia Subsystem (IMS); Stage 2 (Release 5). 3GPP, June 2006.
[10] J. Rosenberg et al.: RFC 3261 - SIP: Session Initiation Protocol. IETF, June 2002.
[11] Web site of IETF P2PSIP WG: http://www.p2psip.org/
[12] Sun Microsystems, Open Cloud, JSR-000240 Specification, Final Release, "JAIN SLEE (JSLEE) 1.1", SUN, 2008.
[13] Sun Microsystems, Oracle Corporation, JSR-000220 Specification, Final Release, "Enterprise JavaBeans, Version 3.0", SUN, May 2006.
[14] Mobicents Open Source JAIN SLEE Server, http://www.mobicents.org
[15] IBM, Microsoft, Specification V 2.0, "Web Services Business Process Execution Language Version 2". OASIS, April 2007.
[16] W3C, "Extensible Markup Language (XML) 1.0 (Fith Edition)", W3C Recommendation, November 2008.
[17] M. Gudgin, et al.: "SOAP Version 1.2 Part 1. Messaging Framework (Second Edition)". W3C, April 2007.
[18] R.H. Glitho, A. Poulin: "A high level service creation environment for Parlay in a SIP environment". IEEE International Conference on Communications, 2002.
[19] R.H. Glitho et al.: "Creating value added services in Internet telephony: an overview and case study on a high-level service creation environment". IEEE Transaction on Systems, Man, and Cybernetics, Part C, November 2003.

[20] J.-L. Bakker, R. Jain: "Next generation service creation using XML scripting languages". IEEE International Conference on Communications, 2002.
[21] E. Christensen et al.: "Web Services Description Language (WSDL) 1.1". W3C, March 2001.
[22] S. Bessler et al.: "An Orchestrated Execution Environment for Hybrid Services". Kommunikation in Verteilten Systemen, vol.II, page.77-88, April 2007, Springer Berlin Heidelberg.
[23] G. Jie et al.: "A Template-based Orchestration Framework for Hybrid Services". AICT '08, Telecommunications, June 2008.
[24] P. Falcarin, C. Venezia: "Communication Web Services and JAIN-SLEE Integration Challenges". International Journal of Web Services Research, Vol. 5, Issue 4, page. 59-78, 2008.
[25] Sun Microsystems, JSR-000112 Specification, Final Release, "J2EE Connector Architecture 1.5", SUN, November 2003

Published in *4th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2011)*, Toulouse, France, pp. 1-8, IEEE, ISBN: 978-1-4577-1192-3

A new Service Description Language as basis for Service Composition in SIP-based Peer-to-Peer infrastructures

A. Lehmann, U. Trick
Research Group for Telecommunication Networks
University of Applied Sciences
Frankfurt/Main, Germany
e-mail: {lehmann, trick}@e-technik.org

W. Fuhrmann
University of Applied Sciences
Darmstadt, Germany
e-mail: w.fuhrmann@fbi.h-da.de

B. Ghita
Centre for Security, Communications, and Network research
University Plymouth, UK
e-mail: bogdan.ghita@plymouth.ac.uk

Abstract—The increasing demand for value added services is a driver for telecommunication companies to find simple and cost efficient solutions for automated service creation, as bespoke service development for small customer groups is not economically viable. The aim of this paper is to propose a peer-to-peer based environment that supports deployment and discovery of value-added services. The identified services can subsequently be aggregated to fulfil more complex individual demands. In this paper a new service description language is introduced to enable the basis for automatic service composition. The proposed approach may be implemented as a peer-to-peer based NGN using SIP signalling.

Keywords-component; Service Composition; NGN (Next Generation Networks); Peer-to-Peer; Session Initiation Protocol (SIP); Service Description Language

I. Introduction

Service provisioning is playing a key role in Next Generation Networks (NGN). The traditional service provisioning approach, relying on the mobile operator as the sole provider, does include a number of advantages, primarily service availability and control. However, the main disadvantage of the approach is the relatively limited level of choice for the consumer. In future, an important discriminating factor when choosing providers is likely to be the range of value-added services (VAS), dictated by the increasing customer demand for services tailored specifically for their needs. To meet these requirements, one of the critical steps is to implement support for openness towards new services. This will lead to a number of new challenges, such as opening the network for external services, beyond the ones from the mobile network provider, or possibilities to build up new services by giving customers facilities to extend them. The Open Mobile Alliance (OMA) specified the Next Generation Service Interface (NGSI) framework [1], in support of developers creating new services that integrate the telecommunication networks functionality. In addition, the Parlay [2] and Global Systems for Mobile Communications Association (GSMA) [3] consortia defined APIs for Telecom third party services. All these APIs are based on Web Services and they offer limited capabilities of the underlying network from a telecommunication provider via a gateway. These APIs were solely specified for centralised telecommunication infrastructures.

In NGN-based telecommunication, SIP (Session Initiation protocol), as an IP-based signalling protocol, plays a major role. SIP is based on a simple request-response interaction model that allows developers to interact with individual protocol messages and SIP can start/manage/tear-down sessions for any media type, be it voice, video or application sharing [4]. SIP is not clearly defined as a client-server architecture, but a hybrid peer-to-peer system implementation. In centralised architectures based on SIP, for example in NGN with IP Multimedia Subsystem (IMS) as core infrastructure, services are typically provided by SIP Application Servers (AS).

Given the likely future demand for diversified services and the limitations of the mobile operator to provide them, a viable alternative is to design a peer-to-peer infrastructure. A broad range of services can easily be offered by a peer-to-peer infrastructure, acting as a service environment, because of its benefits in a number of areas, from scalability to diversity. In such a peer-to-peer environment, the services may be provided by any peer, which would be acting as a distributed server. While appealing in terms of its potential offer, such an approach does raise a number of additional problems in relation to the discovery, establishment, combination and management of services. Unlike the single provider scenario, the discovery of new services in peer-to-peer environments is not automatic or controlled, but distributed and uncoordinated. In relation to this, service composition requires that basic services may be reused.

The distributed services facilitated by the peers will be named as atomic services and the composition of atomic services will be named molecular service. A service composition can be explained as follows: given a pool of atomic services A_i, $i=1,N$, a molecular service M_x can be created by combining a subset of the given atomic services A_j. Further, another molecular services M_y can be established using a different subset A_k of atomic services.

The concept of atomic services is not novel, given that the current NGN do support such functionality, the use of URI (Uniform Resource Identifier), with integrated searching and access for the users. In contrast, supporting search for atomic as well as for molecular services in the peer-to-peer (P2P) infrastructure requires a mechanism to aggregate services automatically, so subscribers have the possibility to search for new service, potentially having the option to describe molecular services [5] [6].

The aim of this paper is to address the issues related to the description and identification of atomic services in order to allow providers and customers to discover newly published atomic services provided by peers and allow providers to combine services to new sets of services, ultimately offering a wider choice of VAS. To address these issues, the peer-to-peer infrastructure has to support publishing and discovery of atomic services, as well as the ability to describe services. The proposed solution is a new Service Description Language that describes uniquely and exhaustively each atomic service, by defining its inputs, outputs, and functionality. A secondary aim of the paper is to optimise the provisioning and composition of distributed VAS for future SIP-based telecommunication networks that are based on a P2P infrastructure, based on the principles of Services Oriented Architecture [7].

The reminder of this paper is structured as follows: the next section presents the proposed architecture and its functionality. Section III describes the Service Composition with an exemplary scenario then section IV discusses the design of a Service Description Language for Service Composition. Section V shows an example of Service Composition based on the developed Service Description Language and section VI concludes the paper.

II. Architecture of SIP-based Peer-to-Peer network

Given its wide deployment, flexibility and ease of use, SIP represents the de facto standard for signalling and communication in modern telecommunication environments. A P2P infrastructure, allowing client-to-client communication, can be established using only the typical SIP network elements (User Agent, Proxy Server, Registrar Server, and Location Server); a proof-of-concept SIP-based infrastructure was already introduced in [8]. The communication mode in such an environment is based on the traditional SIP interaction, with extensions of messaging and entities to allow full support for participating peers. As shown in Figure 1, the SIP Registrar and SIP Proxy servers are needed to lookup the location of the called party (SIP User Agent B). Therefore the temporary SIP URI is resolved to route the initial SIP request and its response (`INVITE` and `200 OK`). It is obvious that the three-way-handshake (by sending `ACK`) is completed directly between the single peers, respectively clients. From this point all data (payload and signalling) will be exchanged in peer-to-peer manner. Such a hybrid P2P infrastructure can be used to provide quick and easy services, especially value-added services.

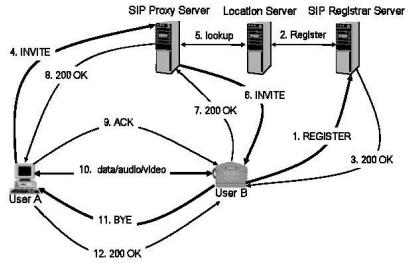


Figure 1. SIP interaction for a hybrid p2p infrastructure

Although SIP may provide the basic required functionality for session management, a complete P2P solution would require further support and entities in order to offer a service similar to the client-server architecture. First a P2P overlay composed of SIP Registrar/Proxy servers has to be created (see Figure 2). The domain for these elements can be resolved by DNS (Domain Name System) [9] [10], or dynamic DNS [11]. Now further SIP network elements can simply register with the overlay network.

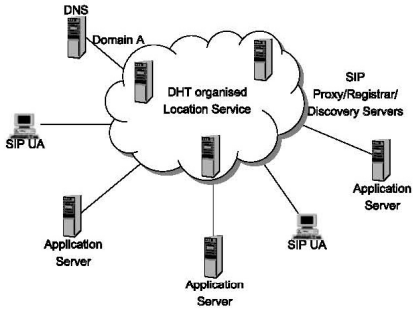


Figure 2. SIP-based P2P network

The location server functionality can be implemented using a Distributed Hash Table (DHT) implementation based on the SIPDHT2 project [12]. The DHT organised location server is holding the data of the registrations from the peers to provide the SIP routing. With the DHT in place,

application servers and media servers can easily be connected to the overlay as peers, ready to provide value-added services [13] [14]. To manage the addressability of application and media servers by their SIP URI, these servers also have to register with the overlay network. Furthermore the authentication of these servers is verified, so that only authorised servers can provide new services. The advantages of this approach are the usage of existing specifications such as SIP or DNS, as well as standard SIP User Agents. Furthermore, the fast lookups, which resolve the temporary SIP URI from the permanent SIP URI, are well provided by the hybrid approach. Further, the architecture allows to register and deregister new services, as demonstrated in the following sections, and also provides service discovery (supported through SIP messages e.g. `NOTIFY`, `SUBSCRIBE`). Due to the P2P concept, the reduction of expenses and scalability will be adopted.

Based on the presented environment, the functionality of the SIP application servers must be extended in order to provide these services to end users. Before providing services, an application server has to register with the network using the `REGISTER` request [8], then each new service has to register [15] [13] [14]. Assuming that an application server has already registered the following URI "as1@domain.co.uk", it is authorised to send the following SIP request (see Figure 3) to the SIP Registrar resolved by DNS to register service X.

```
REGISTER sip: domain.co.uk SIP/2.0
To: <sip:serviceX.as1@domain.co.uk>;tag=1234
Contact: <sip:serviceX.as1@88.88.88.88>
Expires: 3600
```

Figure 3. SIP service registration request

This registration process uses an URI that originates from the subdomains as they are used in DNS, where the service name is used as a subdomain of the application server "as1".

After registration, the new service is reachable by its URI and subscribers can make use of it. By using this model, registered application servers can provide any number of services. To support this functionality, the `Expires` SIP header field sets the period of time for the validity of the registration. As a result, the application server has to register the service with the underlying network in predefined intervals of time in order to make it accessible (here in one hour time slots). With the same SIP request, the service can also be deregistered from the network by sending the SIP header field `Expires` with the value 0. This mechanism fully supports openness for new services.

In addition to the described mechanisms, the overlay network has to be extended in order to construct an architecture that supports publishing and discovery of p2p services. Recently, the Internet Engineering Task Force (IETF) has standardised SIP and several extensions, particularly the SIP event framework [16], as the architecture for status delivery from and to any host in IP-networks and the SIP extension for event state publishing [17]. As part of the architecture, new services will be published using the SIP `PUBLISH` method. The body of this SIP method contains the service description (such as the service name, human readable service description, service id, and service URI) specified using XML [18]. The `PUBLISH` method is sent to the overlay network, where the SIP Proxy and Registrar servers are extended by a so-called SIP event server regarding to [19] to support the publishing and discovery of services. All information belonging to the events is hosted by the SIP event server, named SIP Discovery server for the remainder of this paper.

The message body of the SIP `PUBLISH` method contains the service description. Two further messages, namely `SUBSCRIBE` and `NOTIFY`, are used to realise the discovery functionality. The SIP messages `SUBSCRIBE` and `NOTIFY` will be used for service query and answering. Figure 4 depicts the service discovery process and the associated messages.

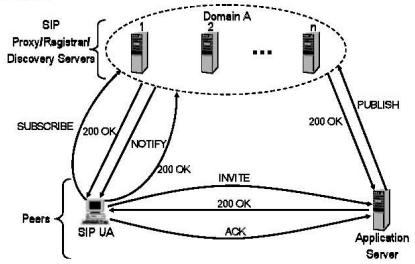


Figure 4. Find, bind and execute paradigm in SIP-based P2P networks

The SIP UA subscriber initiates the process by sending a `SUBSCRIBE` service query message to subscribe to an event and receives `NOTIFY` for the initial notification (which contains the service information). Depending on the lifetime specified within the subscription, subsequent `NOTIFY` messages might be created. In case of a lifetime of zero, the SIP Discovery server merely sends back all available and matching services at this point of time. If the lifetime is greater than zero, the exchange establishes an availability subscription to future services [18] [19].

The resulting architecture, presented in Figure 5, matches the *find, bind and execute* paradigm of the SOA. A so called service triangle could be identified and integrated into a general NGN strata view, as shown in Figure 5 [7]. The SIP Discovery servers are representing the service directory, the service provider is the AS, and the service customer is the SIP User Agent.

The depicted architecture includes all the necessary components and associated interaction to support the proposed service composition, therefore is the basis for Service Composition, presented in the next section. The presented architecture had already implemented for verification.

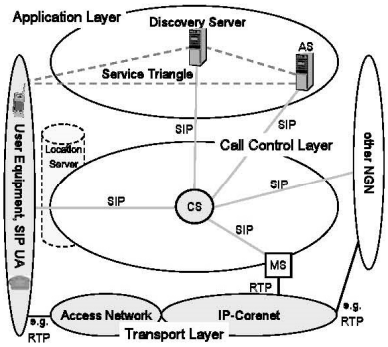


Figure 5. Service triangle in general NGN strata view

III. Service Chaining

The framework described in section II simplifies significantly the combination of distributed services. The following will present a simple method to realise invocation of combined services.

The SIP `Route` Header [8] will be used for invoking the services regarding to [20]. The SIP `Route` header holds a set of SIP URIs, each of them representing the addresses of the atomic services to be combined. Based on the formalised description a set of atomic services $A_k, k=1,N$ are invoked in the sequence they will appear separated by comma in the `Route` header ($R=A_1,A_2,A_3$) followed by the atomic service declared in the SIP request line L.

The concept introduced by the MultiService Forum (MSF) [21] is based on a centralised architecture referred to IMS. The Service Broker is defined as the central entity that manages service identification and routing. This approach was adjusted for realising service compositions in SIP-based peer-to-peer networks[7]. The following will illustrate the service chaining with the extension of manipulating media streams.

A typical scenario of service composition can be an Audio/Video conference with an embedded news ticker which presents text-based live information. In the proposed distributed service architecture, every atomic service is addressable by a permanent SIP URI, such as audio.conf@p2pnet.de, while composed services can be achieved by chaining the SIP URIs. The resulting service could be described by the following SIP message (note only the relevant header fields are shown) presented in Figure 6.

```
INVITE sip: videoconference.bt@99.99.99.99 SIP/2.0
…
Route: <sip: audio.conf@77.77.77.77;lr>,
       <sip: newsticker.bbc@88.88.88.88;lr>
…
```

Figure 6. SIP service chain

The SIP message would be routed using the `Route` headers, as shown in Figure 7.

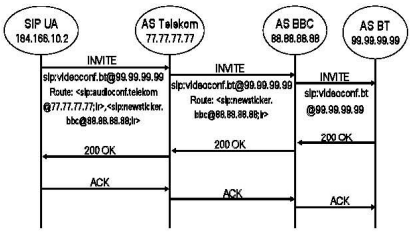


Figure 7. Exemplary service composition with SIP Route Header

As depicted in Figure 7, the SIP message `INVITE` is forwarded from one service to another, the services themselves may convert parts of the SIP message, and the news ticker service might modify the SIP body of the initial message. For the purpose of embedding the news ticker into the video stream, the SDP media description part, containing video information (see Figure 8), has to be modified by the news ticker service, so that all video data will be forwarded to the news ticker. As a result, the news ticker service can overlay the ticker information into the video data. This can be achieved by modifying the corresponding information from the SIP body as follows.

```
…
v=0
o=client 1234 0 IN IP4 184.166.10.2
s=ServiceCombination
t= 0 0
c=IN IP4 184.166.10.2
m=audio 1000 RTP/AVP 0 8 3
m=video 2000 RTP/AVP 34
…
```

Figure 8. SDP for an atomic video service

The submitted IP address given by the Connection Data parameter `c` and the submitted port given by the Media Description parameter `m` (which in Figure 8 has the value 2000, indicating that the video will run on port 2000) specify the socket the subscriber will expect the video data. The news ticker service adds a new `c` parameter to the SIP body in front of the `m` parameter (video) with its own IP address so that he can receive the video data. Figure 9 will present the modified SIP body.

```
…
v=0
o=client 1234 0 IN IP4 184.166.10.2
s=ServiceCombination
t= 0 0
c=IN IP4 184.166.10.2
m=audio 1000 RTP/AVP 0 8 3
c=IN IP4 88.88.88.88
m=video 2000 RTP/AVP 34
```

Figure 9. Modified SDP for a molecular service

By modifying the SIP message structure as described, the video payload can be routed as intended and the combining of services can be easily implemented. This approach takes advantage of the P2P architecture benefits, such as scalability, cost reduction, fault tolerance, and it provides full support for standardised SIP network elements, such as SIP User Agents and SIP Proxy/Registrar servers.

The basis for service composition is a new service description language that is explained in the next chapter. This service description language supports the requirement for appropriate designing molecular services and defines the further requirements, as for example temporal relations between atomic or molecular services.

IV. Service Description Language

A well defined service description language is the critical element of automated service composition for a number of reasons. Firstly, the SIP Discovery Server, which creates the service composition, needs to have knowledge about the services and their interfaces, so that it can combine them. Secondly, the service description needs to be precise enough for the service to be uniquely identifiable. Finally, the description language has to be machine readable and interpretable in order to streamline the service composition underlying process.

The proposed service description language consists of five constructs: *service Id*, *temporal composition*, *service goal*, *functional properties*, and *non-functional properties*. The *service Id* is an identifier for an atomic service represented by its permanent SIP URI. The *temporal composition* describes how different atomic or other molecular services are combined to produce the defined service; it is based on all temporal relations between the single services, as formalised in [22]. According to Allen [23], relations between two intervals can be divided into 13 different relations using the `before`, `during`, `meets`, `finishes`, `overlaps`, `starts`, and `equals` connections. Figure 10 shows seven of them. The other six relations are derived through inversion of these, apart from the `equals` one. The linkage between the service and the temporal relations is defined recursively (see Figure 11). An example presented in the next section will describe the *temporal composition* more deeply. The *service goal* is a set of keywords describing the functionality so that humans can imagine what the service will offer. For example a service goal with the key words “wake up” and “instant message” might describe a service that will send an instant message as a wake up message. *Non-functional properties* are, for example, cost or security. They are used to limit the space of compositions that fulfil the service request and to rank the generated set of compositions, such as ranking higher a service with lower costs and higher reliability.

The *functional properties* are the *conditions* (pre-, continuous and post-condition), *connectors* (in-, through- and output), *media objects* (source, sink and filter), *communication* (a/synchronous) and *interaction* (non-/real-time). The *conditions* are properties that must be considered before, during or after service processing; an example of a condition is that, prior to any interaction, the SIP session must be established.

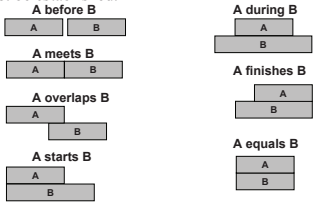


Figure 10. Possible temporal relations

The *connectors* are major elements for aggregating services. All in-, out- and throughputs offering attributes to identify if the *connectors* are for the media stream of the originating party (the party who initiated the service invocation) or if they belong to the terminating party. *Connectors* also may be optional, this means they can be utilised if they are needed, for example in conferences up to 10 participants. Furthermore, the *connectors* are linked to *media types*. These *media types* are derived from human senses (hearing, sight, smell, touch, taste) also another *media type* is defined which is not sensually imperceptible. Four further senses (pain, balance, proprioception and temperature) are known, which will be assigned to the sense touch, only for simplification. These *media types* will cover all possibilities for transferable media in telecommunication. Table I gives an overview of the media types and their characteristics regarding to [22].

TABLE I. MEDIA TYPES AND THEIR CHARACTERISTICS

Media Type	Characteristics
Hearing	sound, speech, music
Sight	video, animation, text, still image, light
Smell	camphorous, fishy, malty, minty, musky, spermous, sweaty, urinous
Touch	pressure, temperature, balance, proprioceptive
Taste	sweet, sour, bitter, spicy, umami
Imperceptible	file, sensor, actuator, trigger, information

The *connectors* are forming ports for linking them with the matching opposite ones. For example Out- and Throughputs can only be linked to In- or Throughputs which will fit, that is they have to be from the same *media type* and must own the same characteristic (such as Hearing and Sound). Ports of different *media types* and different characteristics are incompatible and therefore cannot be combined. The characteristics shown in Table I are

specifications of the *media types*, so they can be assigned more precisely. This is essential in communication networks, because different media type characteristics may have different bandwidths/sampling rates; for example, standard sampling rate for speech is 8 kHz in contrast to music in CD-quality, which is 44.1 kHz. In telephony special Codecs exists for these different demands. The grading of quality aspects within one characteristic e.g. speech is a *non-functional property*. In contrast, the imperceptible media type is characterised by different technical aspects or formats. This is required for aggregating the same characteristics of imperceptible *media types*.

The *media objects* constructs include media *sources*, *sinks* or *filters*. The *sources* output data streams and the *sinks* receive data streams of concrete media types. *Filters* combine the characteristics of *sources* and *sinks* and they are mostly used to manipulate data streams. For this purpose, the *filters* are divided into different functionalities namely *input/output-selector*, *synchroniser* (synchronise different *media types* such as speech and video), *inter-media* and *intra-media*. The *input-selector* is a device that selects one of several input *media types* and forwards the selected *media type* as output. On the other hand an *output-selector* takes a single input *media type* and forwards it to a chosen output from several. Both selector objects are controlled by an input *media type* (e.g. an imperceptible media input like sensor controls hearing *media types*). *Inter-media filters* are processing different *media types* however *intra-media filters* are only processing one *media type*. These two media *filters* can be divided into *converter* (for example Text-to-Speech), *splitter* (for example separate subtitle from video) and *merger* (for example bundle speech and video to a file as output) for *inter-media filters* and *intra-media filters* can be divided into *mixer* (for example video conferences or text chat), *multiplier* (duplicate the media type) and *modifier* (for example change the size of a video).

Also general information has to be defined for all *media objects,* this is the form of *communication* (for example one-to-one, many-to-one) and last but not least the way of *interaction* has to be defined. For example a real-time service will be a normal voice call in contrast to instant messaging which is a non-real-time service.

Figure 12 gives an overview of the basic structure for the service description language, which is based on XML. The *temporal composition* defines how services (atomic and molecular services) are arranged. This is done recursively (see Figure 11).

```
<Service>
 <ServiceID id="serviceComposition"/>
 ...
 <TemporalComposition>
  <ServiceID id="atomicService">
   <TimeRelation relation="before">
    <ServiceID id="molecularService">
     <TimeRelation relation="equals">
      ...
     </TimeRelation>
    </ServiceID>
   </TimeRelation>
  </ServiceID>
 </TemporalComposition>
</Service>
```

Figure 11. Temporal Composition

Figure 11 shows that also molecular services might be contained in the *temporal composition*. Every molecular service holds its own *temporal composition*. This means that *temporal compositions* can be nested.

Based on this service description language compositions can be arranged so that machines can interpret them.

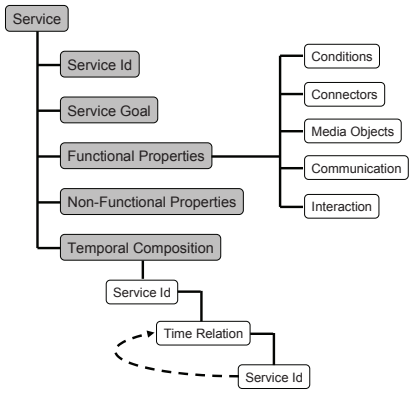


Figure 12. Structure of the Service Description Language

The following section will present an example to clarify the usage and construction of the service description language.

v. Example

The service scenario presented in section III is a good example to demonstrate how service composition is described using the service description language.

First it is necessary to take a look at the service descriptions of the three atomic services (audioconference, videoconference and news ticker). Only the relevant properties of these services will be discussed in the following subsections. The *service Ids* are audio.conf@p2pnet.de for the audio conference service, videoconference.bt@p2pnet.de for video conference service and newsticker.bbc@p2pnet.de for the last service. The properties of the audio conference service are shown in Figure 13.

```
<ServiceID id="audio.conf@p2pnet.de"/>
<FunctionalProperties>
 <Connectors>
  <Input id="1" optional="false" case="originating">
   <hearing audtion="speech"/>
```

```
 </Input>
     ...          further inputs
 <Output id="1" optional="false" case="originating">
  <hearing audition="speech"/>
 </Output>
     ...          further outputs
</Connectors>
<MediaObjects object="filter">
 <Intra-Media type="mixer">
  <InputIDs> 1, 2, ..., m </InputIDs>
  <OutputIDs> 1, 2, ..., n </OutputIDs>
 </Intra-Media>
 <CommunicationForm>
  <Many-Many inParties="m" outParties="n"/>
 <CommunicationForm>
</MediaObjects>
<Communication synchronisity="synchronous"/>
<Interaction reaction="real-time"/>
<FunctionalProperties>
```

Figure 13. Audio Conference Service Description

The only differences between the description of the video and the audio conference services are in the *connectors*. The *media type* of the audio conference service is hearing and the *media type* for the video conference service is sight with characteristic video.

The properties of the news ticker service are presented in Figure 14.

```
<ServiceID id="newsticker.bbc@p2pnet.de"/>
<FunctionalProperties>
 <Connectors>
  <Input id="1" optional="false" case="terminating">
   <sight vision="video"/>
  </Input>
  <Output id="1" optional="false" case="originating">
   < sight vision="video"/>
  </Output>
 </Connectors>
 <MediaObjects object="filter">
  <Intra-Media type="modifier">
   <InputIDs> 1 </InputIDs>
   <OutputIDs> 1 </OutputIDs>
  </Intra-Media>
  <CommunicationForm>
   <One-One/>
  <CommunicationForm>
 </MediaObjects>
<Communication synchronisity="synchronous"/>
 <Interaction reaction="real-time"/>
<FunctionalProperties>
```

Figure 14. News Ticker Service Description

In this scenario, service composition is defined by the *temporal composition* properties. The audio conference *A* service and the news ticker service *B* have to run in parallel. Translated in the *temporal composition* syntax defined earlier on, *A = B* (*A equals B*). During these two services the video conference service is running (see Figure 15), because the outgoing media streams are connected to the news ticker service so that the news tickers data can be embedded into the media streams of the video conference (see Figure 16).

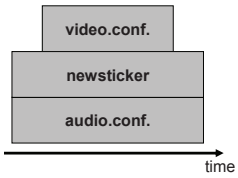


Figure 15. Temporal relation between services

As shown in Figure 16 the incoming video data will be handled first by the video conference service and the resulting data will be delivered to the news ticker. Therefore the news ticker must have the ability to receive data before the video conference service can transmit the data to it. So the news ticker has to be started before the video conference service, and the audio conference service should run in parallel.

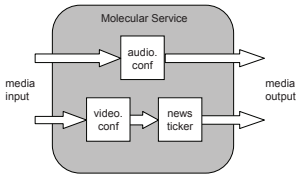


Figure 16. Media processing

The temporal composition within the service description language is shown in Figure 17. The resulting service invocation presented in Figure 7 can easily derived from the temporal composition.

```
<TemporalComposition>
 <ServiceID id="videoconference.bt@p2pnet.de">
  <TimeRelation relation="during">
   <ServiceID id="audio.conf@p2pnet.de">
    <TimeRelation relation="equals">
     <ServiceID id="newsticker.bbc@p2pnet.de"/>
    </TimeRelation>
   </Service>
  </TimeRelation>
 </Service>
</TemporalComposition>
```

Figure 17. Temporal Composition

The synchronisation between the audio and video streams is not handled by the service, because there is no synchronizing media object included. The synchronicity of media streams could be managed by the terminal equipments.

This short example details on how to use the proposed service description language to form molecular service compositions out of existing atomic services. The following

chapter is concluding this article and will discuss further steps to implement this approach into a prototypical environment for validation.

VI. Conclusion

Automated service composition for value-added services is the most important means of innovation in peer-to-peer based telecommunication networks. In particular, automated service composition enhances the economy and allows service creation for small customer groups.

The presented approach empowers customers to use specialised services in a cost efficient way. The presented architecture was already tested under laboratory conditions. First tests also have shown that automatic service composition is possible.

Next steps to go are development and implementation of an adequate algorithm to automatically build service compositions based on the presented service description language. Furthermore the service composition process has to be validated. This will be done by specifying a set of atomic services, which have to be combinable. Also the reusability and modularity of atomic services has to be analysed. This means further research on atomic services and their specialisations has to be done.

References

[1] Open Mobile Alliance: Application Programming Interfaces (APIs), Available at: www.openmobilealliance.org/comms/pages/oma_apis.htm [Accessed 16 June 2011].

[2] 3GPP: 3GPP Specification series TS 29.199-01 up to TS 29.199-22, Available at: www.3gpp.org/ftp/Specs/html-info/29-series.htm [Accessed 16 June 2011].

[3] GSMA: GSMA OneAPI, Available at: www.gsmworld.com/oneapi/ [Accessed 16 June 2011].

[4] Banerjee, N.; Chakraborty, D.; Mittal, S.: "Service Delivery Platforms: Developing and Deploying Converged Multimedia Services", Syed, A.A; Ilyas, M., Auerbach Publications p.1 – 28 ISBN 978-1-4398-0089-8

[5] Silva, E; Pires, L; van Sinderen, M.: "An Algorithm for Automatic Service Composition", 1st International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing, ICSOFT 2007, Barcelona, Spain, July 2007.

[6] Zhao, Z.; Laga, N.; Crespi, N.: "The Incoming Trends of End-User driven Service Creation", First International ICST Conference, DigiBiz 2009, London, UK, June 2009.

[7] Lehmann, A; Trick, U.; Fuhrmann, W.: "SOA-basierte Peer-to-Peer-Mehrwertdienstebereitstellung", 14th VDE/ITG Mobilfunktagung, Osnabrück, May 2009.

[8] Rosenberg, J.; Schulzrinne, H.; Cammarillo, G.; Johnston, A.; Peterson, J.; Sparks, R.; Handley, M.; Schooler, E.: "SIP: Session Initiation Protocol", RFC 3261, IETF, June 2002.

[9] Mockapetris, P.: "Domain Names – Concepts and Facilities", RFC 1034, IETF, November 1987.

[10] Mockapetris, P.: "Domain Names – Implementation and Specification", RFC 1035, IETF, November 1987.

[11] Vixie, P.; Thomson, S.; Rekhter, Y.; Bound, J.: "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, IETF, April 1997.

[12] Fandrianto, A.: "SIPDHT: An Open Source Project for P2PSIP", Implementation report, Cisco Systems, April 2007.

[13] Lehmann, A.; Eichelmann, T.; Trick, U.: "Neue Möglichkeiten der Dienstebereitstellung durch Peer-to-Peer-Kommunikation", 13th VDE/ITG Mobilfunktagung, Osnabrück, May 2008.

[14] Lehmann, A.; Fuhrmann, W.; Trick, U.; Ghita, B.: "New possibilities for the provision of value-added services in SIP-based peer-to-peer networks", University of Wrexham, Proc. of SEIN, November 2008.

[15] Schmidt, H.; Guenkova-Luy, T.; Hauck, F.J.: "Service Location using the Session Initiation Protocol (SIP)", International conference on Networking and Services (ICNS), Silicon Valley, CA, July 2006.

[16] Roach, A.: "Session Initiation Protocol (SIP) Specific Event Notification", RFC 3265, IETF, June 2002.

[17] Niemi, A.: "Session Initiation Protocol (SIP) Extension for Event State Publication", RFC 3903, IETF, October 2004.

[18] Rahman, M.; Buford, J.: "Service Discovery using Session Initiation Protocol (SIP)", United States Patent Application Publication, June 2006.

[19] Trossen, D.; Pavel, D.: "Service discovery & availability subscriptions using the SIP event framework", IEEE International Conference on Communications, Seoul Korea, May 2005.

[20] MSF-IA-SIP.006-FINAL: "Implementation Agreement for SIP interface between Service Broker and Application Server", Implementation Agreement, MultiService Forum, April 2005.

[21] MSF (MultiService Forum): http://www.msforum.org [Accessed 5 July 2011].

[22] Lehmann, A.; Fuhrmann, W.; Trick, U.; Ghita, B.: "A new approach to classify and describe telecommunication services", Proc. of SEIN 2009, University of Applied Sciences Darmstadt, Germany, November 2009.

[23] Allen, J.: "Maintaining knowledge about temporal intervals", Communications of the ACM, ACM, 1983.